```
TITLE 'COLLEEN CALCULATOR, BY C SHAW'
                ; 1=> ASSEMBLE THIS SECTION, 0=> THIS STUFF
      0
ATARI CALCULATOR CARTRIDGE  COPYRIGHT 1979
WORK STARTED 2/20/79
PROGRAM STARTED 3/14/79

ING SYSTEM EQUATES
= $E456 CENTRAL INPUT OUTPUT ROUTINE
=       $E45C           ;SET SYSTEM TIMERS ROUTINE

ND CODES FOR IOCB
= 3             OPEN FOR INPUT/OUTPUT
= 7     $B      GET CHARACTER(S)
        $C      ;PUT CHARACTER(S)
                ;CLOSE DEVICE

EUL OPERATION
                ;NUMBER OF BYTES PER IOCB
```

6/13 2:45 save

9K

(some stuff

(wing

DM

COLLEEN CALCULATOR, BY C SHAW

6/13 2:45 save

9K

(some stuff already removed)
(using conditional assembly

DMS

```
                        .TITLE  'COLLEEN CALCULATOR, BY C SHAW'
0000                    ASMBL   =       0               ;1=> ASSEMBLE THIS SECTION, 0=> THIS STUFF HAS BEEN REMOVED
                        ;
                        ;               ATARI CALCULATOR CARTRIDGE  COPYRIGHT 1979
                        ;               WORK STARTED 2/20/79
                        ;               PROGRAM STARTED 3/14/79
                        ;
                        ; OPERATING SYSTEM EQUATES
                        ;
E456                    CIOV    = $E456 CENTRAL INPUT OUTPUT ROUTINE
E45C                    SETVBV  =       $E45C           ;SET SYSTEM TIMERS ROUTINE
                        ;
                        ; COMMAND CODES FOR IOCB
0003                    OPEN    = 3             OPEN FOR INPUT/OUTPUT
0007                    GETCHR  = 7             GET CHARACTER(S)
000B                    PUTCHR  =       $B              ;PUT CHARACTER(S)
000C                    CLOSE   =       $C              ;CLOSE DEVICE

0001                    SUCCES  = $01   SUCCESSFUL OPERATION
0010                    IOCBSZ  =       16              ;NUMBER OF BYTES PER IOCB
                        ;
                                *=$11
0011                    BRKKEY  *=*+1           ;BREAK KEY FLAG
                                *=$52
0052                    LMARGN  *=*+1           ;LEFT MARGIN (0 MIN.)
0053                    RMARGN  *=*+1           ;RIGHT MARGIN (39 MAX.)
0054                    ROWCRS  *=*+1   CURSOR COUNTERS
0055                    COLCRS  *=*+2
                        ;
                                *=$22A
022A                    CDTMF3  *=*+1           ;COUNT DOWN TIMER 3 FLAG
                                *=$2F0
02F0                    CRSINH  *=*+1           CURSOR INHIBIT (00 = CURSOR ON)
                        ;
                        ;       NOTE : THE ENTIRE IOCB DEFINITIONS HAVE BEEN MODIFIED
                        ;
                                *=$340
0340                    IOCB    *=*     I/O CONTROL BLOCKS
0340                    ICHID   *=*+1   HANDLER INDEX NUMBER (FF = IOCB FREE)
0341                    ICDNO   *=*+1   DEVICE NUMBER (DRIVE NUMBER)
0342                    ICCOM   *=*+1   COMMAND CODE
0343                    ICSTA   *=*+1   STATUS OF LAST IOCB ACTION
0344                    ICBAL   *=*+1   BUFFER ADDRESS LOW BYTE
0345                    ICBAH   *=*+1
0346                    ICPTL   *=*+1   PUT BYTE ROUTINE ADDRESS - 1
0347                    ICPTH   *=*+1
0348                    ICBLL   *=*+1   BUFFER LENGTH LOW BYTE
0349                    ICBLH   *=*+1
034A                    ICAX1   *=*+1   AUXILIARY INFORMATION FIRST BYTE
034B                    ICAX2   *=*+1
034C                    ICSPR   *=*+4   FOUR SPARE BYTES
                        ;
                        ;                               FLOATING POINT SUBROUTINES
                        ;
0006                    FPREC   =       6               ;FLOATING PT PRECISION (# OF BYTES)
                        ;                               IF CARRY USED THEN CARRY CLEAR => NO ERROR, CARRY SET => ERROR
D800                    AFP     =       $D800           ;ASCII->FLOATING POINT (FP)
                        ;                               INBUFF+CIX -> FR0, CIX, CARRY
D8E6                    FASC    =       $D8E6           ;FP -> ASCII      FR0-> LBUFF  (INBUFF)
D9AA                    IFP     =       $D9AA           ;INTEGER -> FP
```

```
D8E6                FASC    =       $D8E6           ;FP -> ASCII        FRO-> LBUFF  (INBUFF)
D9AA                IFP     =       $D9AA           ;INTEGER -> FP
```

COLLEEN CALCULATOR, BY C SHAW

```
                    ;                               0-$FFFF (LSB,MSB) IN FRO,FRO+1->FRO
D9D2                FPI     =       $D9D2           ;FP -> INTEGER    FRO -> FRO,FRO+1, CARRY
DA60                FSUB    =       $DA60           ;FRO <- FRO - FR1  ,CARRY
DA66                FADD    =       $DA66           ;FRO <- FRO + FR1  ,CARRY
DADB                FMUL    =       $DADB           ;FRO <- FRO * FR1  ,CARRY
DB28                FDIV    =       $DB28           ;FRO <- FRO / FR1  ,CARRY
DD89                FLDOR   =       $DD89           ;FLOATING LOAD REGO    FRO <- (X,Y)
DD8D                FLDOP   =       $DD8D           ;    "      "  "      FRO <- (FLPTR)
DD98                FLD1R   =       $DD98           ;    "      "   REG1   FR1 <- (X,Y)
DD9C                FLD1P   =       $DD9C           ;    "      "    "     FR1 <- (FLPTR)
DDA7                FSTOR   =       $DDA7           ;FLOATING STORE REGO (X,Y) <- FRO
DDAB                FSTOP   =       $DDAB           ;    "      "    "   (FLPTR)<- FRO
DDB6                FMOVE   =       $DDB6           ;FR1 <- FRO
DD40                PLYEVL  =       $DD40           ;FRO <- P(Z) = SUM(I=N TO 0) (A(I)*Z**I)     CARRY
                    ;                               INPUT: (X,Y) = A(N),A(N-1)...A(0)  -> PLYARG
                    ;                                      ACC   = # OF COEFFICIENTS = DEGREE+1
                    ;                                      FRO   = Z
DDC0                EXP     =       $DDC0           ;FRO <- E**FRO = EXP10(FRO * LOG10(E))  CARRY
DDCC                EXP10   =       $DDCC           ;FRO <- 10**FRO  CARRY
DECD                LOG     =       $DECD           ;FRO <- LN(FRO) = LOG10(FRO)/LOG10(E)   CARRY
DED1                LOG10   =       $DED1           ;FRO <- LOG10 (FRO)    CARRY

                    ;                               THE FOLLOWING ARE IN BASIC CARTRIDGE:
                    ;SIN    =       $BD81           ;FRO <- SIN(FRO)  DEGFLG=0 =>RADS,  6=>DEG.   CARRY
                    ;COS    =       $BD73           ;FRO <- COS(FRO)    CARRY
                    ;ATAN   =       $BE43           ;FRO <- ATAN(FRO)   CARRY
                    ;SQR    =       $BEB1           ;FRO <- SQUAREROOT(FRO)   CARRY

                    ;FLOATING POINT ROUTINES ZERO PAGE (NEEDED ONLY IF F.P. ROUTINES ARE CALLED)
                            *=$D4
00D4                FRO     *=*+FPREC               ;FP REGO
00DA                FRE     *=*+FPREC
00E0                FR1     *=*+FPREC               ;FP REG1
00E6                FR2     *=*+FPREC
00EC                FRX     *=*+1                   ;FP SPARE
00ED                EEXP    *=*+1                   ;VALUE OF E
00EE                NSIGN   *=*+1                   ;SIGN OF #
00EF                ESIGN   *=*+1                   ;SIGN OF EXPONENT
00F0                FCHRFLG *=*+1                   ;1ST CHAR FLAG
00F1                DIGRT   *=*+1                   ;# OF DIGITS RIGHT OF DECIMAL
00F2                CIX     *=*+1                   ;CURRENT INPUT INDEX
00F3                INBUFF  *=*+2                   ;POINTS TO USER'S LINE INPUT BUFFER
00F5                ZTEMP1  *=*+2
00F7                ZTEMP4  *=*+2
00F9                ZTEMP3  *=*+2
00FB                RADFLG  *=*+1                   ;0=RADIANS, 6=DEGREES
00FC                FLPTR   *=*+2                   ;POINTS TO USER'S FLOATING PT NUMBER
00FE                FPTR2   *=*+2

                    ;FLOATING PT ROUTINES' NON-ZERO PAGE RAM (NEEDED ONLY IF F.P. ROUTINES CALLED)
                            *=$57E
057E                LBPR1   *=*+1                   ;LBUFF PREFIX 1
057F                LBPR2   *=*+1                   ;LBUFF PREFIX 2
0580                LBUFF   *=*+128                 ;LINE BUFFER
05E0                PLYARG  =       LBUFF+$60       ;POLYNOMIAL ARGUMENTS
05E6                FPSCR   =       PLYARG+FPREC
05EC                FPSCR1  =       FPSCR+FPREC
05E6                FSCR    =       FPSCR
05EC                FSCR1   =       FPSCR1
```

COLLEEN CALCULATOR, BY C SHAW

```
                        ;
                                                COLLEEN REGISTER EQUATES
D200                    AUDF1   =       $D200   ;SOUND REG 1 FREQUENCY
D201                    AUDC1   =       AUDF1+1 ;SOUND REG 1 CONTROL
D20A                    RANDOM  =       AUDF1+10        ;8 BIT RANDOM NUMBER
```

UNIVERSAL EQUATES

```
                        ;
                        ;                               CIO COMMANDS
0004                    INPUT   =       4               ;AUX1 ON OPEN
0008                    OUTPUT  =       8               ;AUX1 ON OPEN
                        ;
                        ;                               SPECIAL CHARS IN ATARI EXTERNAL ASCII
001B                    ESC     =       $1B             ;ESCAPE
001C                    UPAROW  =       ESC+1           ;UP ARROW (CONTROL CHAR)
001D                    DNAROW  =       ESC+2           ;DOWN
001E                    LFAROW  =       ESC+3           ;LEFT
001F                    RTAROW  =       ESC+4           ;RIGHT
007D                    CLS     =       $7D             ;CLEAR SCREEN
007E                    BACKSP  =       CLS+1           ;BACKSP
007F                    TAB     =       CLS+2
009B                    CR      =       $9B             ;CARRIAGE RETURN
009C                    DELLIN  =       CR+1            ;DELETE LINE
009D                    INSLIN  =       CR+2            ;INSERT LINE
009E                    CLRTAB  =       CR+3            ;CLEAR TAB
009F                    SETTAB  =       CR+4            ;SET TAB
00FE                    DELCHR  =       $FE             ;DELETE CHAR
00FF                    INSCHR  =       DELCHR+1        ;INSERT CHAR


                        ;                               FP PACKAGE EQUATES FOR SIN, COS, ATAN, AND SQR ROUTINES ETC
000B                    NATCF   =       $B
DA51                    INTLBF  =       $DA51           ;FOR FP -> ASCII CONVERSION

D920                    XEFORM  =       $D920           ;FOR FP -> ASCII CONVERSION
DE95                    XFORM   =       $DE95
DF6C                    FHALF   =       $DF6C
DFAE                    ATCOEF  =       $DFAE
DFEA                    FP9S    =       $DFEA
DFF0                    PIOV4   =       $DFF0
0006                    NSCF    =       6
```

CALCULATOR EQUATES

```
                        ;
0001                    LMARG   =       1               ;LMARGN VALUE
0026                    RMARG   =       38
0026                    LINLEN  =       38              ;LENGTH OF LINE ON SCREEN

0016                    ROWCMD  =       22              ;ROWCRS FOR COMMANDS
0014                    COLCMD  =       20              ;COLCRS FOR COMMANDS
0001                    ROWSTT  =       1               ;ROW # FOR STATUS
0005                    ROWREG  =       5               ;ROW FOR STACK, MEM REGS
0010                    ROWSCR  =       16              ;TOP ROW FOR SCROLLING

0000                    SIOCB   =       0*IOCBSZ        ;SCREEN IOCB #  (SET UP BY OS)
0010                    KIOCB   =       1*IOCBSZ        ;KEYBOARD IOCB #
0020                    PIOCB   =       2*IOCBSZ        ;PRINTER IOCB #
0030                    TIOCB   =       3*IOCBSZ        ;TEMP IOCB # (USED FOR FILE I/O)

000D                    TOKCLN  =       TOKEND-TOKCHR-1         ;LENGTH OF TOKCHR-1
008F                    SLASH   =       STAR+1
0090                    PLUS    =       STAR+2
0091                    MINUS   =       STAR+3
0092                    LPAR    =       STAR+4
0093                    RPAR    =       STAR+5
0094                    EQUAL   =       STAR+6
0095                    LPAD    =       STAR+7
0096                    NUMBER  =       STAR+8

000E                    PSPEC   =       14      ;PRIORITY OF SPECIAL 0-VAR FNS (PRINT, ETC.)
000D                    PHIGH   =       13      ;PRIORITY OF SINGLE VAR FNS.
000A                    PSPEC2  =       10      ;SPECIAL 2-VAR FNS
0009                    PPOWER  =       9       ;POWER, ROOT
0008                    PTIMES  =       8       ;* /
0007                    PPLUS   =       7       ;+ -
0006                    PAND    =       6
0005                    POR     =       5
0002                    PLRPAR  =       2       ;( )
0001                    PEQUAL  =       1       ;=
0000                    PLPAD   =       0       ;BOTTOM OF STACK SYMBOL

0010                    NUMLEN  =       16              ;LENGTH OF NUMBER IN ASCII FORMAT
002A                    FPSLEN  =       42              ;LENGTH OF FPSTK IN FP NUMBERS
0100                    OPSLEN  =       256             ;LENGTH OF OPSTK IN OPERANDS
0064                    MEMLEN  =       100             ;LENGTH OF MEMORY AREA IN FP NUMBERS
0026                    TOKLEN  =       LINLEN          ;LENGTH OF TOKBUF IN CHARS
0400                    PRGLEN  =       1024            ;PROGRAM MEMORY LENGTH IN BYTES
000F                    PC1MAX  =       PRGMEM/256+3            ;MAX PC+1 VALUE
0006                    SPCLEN  =       SPCEND-SPCTBL-1         ;LENGTH OF SPCTBL - 1

                        ;GRADON =       12              ;DEGREE FLAG SETTING FOR GRAD

                        ;                               COLUMN NUMBERS FOR LINE 0 STATUS DISPLAY
0006                    DALG    =       3+3
000B                    DDEG    =       8+3
0010                    DDEC    =       13+3
0018                    DBITS   =       21+3
001E                    DFIX    =       33-6+3
0020                    DOFF    =       35-6+3
```

RAM PAGE ZERO

```
                        ;
                                *=$80
0080                    ZROPG
0080                    LFRT    *=*+1           ;1 => LEFT NIBBLE, 0=> RIGHT NIBBLE (USED BY LDNIB)
0081                    TOKCOD  *=*+1           ;TOKEN CODE
0082                    TOKPTR  *=*+2           ;POINTER TO NEXT TOKBUF LOC
0084                    TOKTMP  *=*+2           ;LAST 0-2 CHARS READ AND SAVED
0086                    TOKTIN  *=*+1           ;INDEX TO TOKTMP (0-2)
0087                    DHOFLG  *=*+1           ;0=> DEC, 16=>HEX, 8=>OCT, 2=>BIN
0088                    KEYCHR  *=*+1           ;CURRENT KEY CHAR (0-?)
0089                    KEYLEN  *=*+1           ;LENGTH OF CURRENT KEY WORD
008A                    KEYLN2  *=*+1           ;KEY LENGTH (MODIFIED BY LDCHR FOR 2 NIBBLE CHARS)
008B                    LDNBSV  *=*+1           ;REG Y SAVED FOR LDNIB
008C                    CLRPTR                  ;FOR RAM CLEAR
008C                    PKPTR   *=*+2           ;POINTER TO PACKED CHAR STRING (USED BY LDNIB)
008E                    KYLFRT  *=*+1           ;LFRT FOR BEGINNING OF WORD
008F                    KEYCNT  *=*+1           ;KEY WORD NUMBER
0090                    KYPTSV  *=*+2           ;KEYPTR SAVED FOR LONGEST MATCH
0092                    KLRSV   *=*+1           ;KYLFRT
0093                    KYCNSV  *=*+1           ;KEYCNT
0094                    KMATCH  *=*+1           ;# OF CHARS WHICH MATCH IN SAVED KEYWORD
0095                    JMPTR1  *=*+2           ;INDEX FOR SUBROUTINE JUMP
0097                    JMPTR2  *=*+2

0099                    RPNALG  *=*+1           ;0=>RPN, 1=>ALG, 2=>ALGN
0001                    ALGP    =       1               ;ALGEBRAIC, OPERATOR PRECEDENCE
0002                    ALGNOP  =       2               ;ALGEBRAIC, SAME PRECEDENCE FOR 2 VAR OPERATORS
009A                    NSEFLG  *=*+1           ;0=> NOEXP UNLESS NECESSARY, 1=> SCIENTIFIC ALWAYS, 2=>ENG
009B                    FIXFLG  *=*+1           ;9=> NOT FIXED, 0-8 => DIGITS AFTER DECIMAL PT.
009C                    PRMFLG  *=*+1           ;1=> PROMPT
009D                    PRNFLG  *=*+1           ;1=>PRINT
009E                    OPFLG   *=*+1           ;1=>PREVIOUS TOKEN WAS AN OPERATOR
009F                    NOPFLG  *=*+1           ;NEW OPFLG  1=> CURRENT TOKEN IS OPERATOR
00A0                    INTERM  *=*+1           ;1=> DISPLAY INTERMEDIATE RESULTS
00A1                    PREVOP  *=*+1           ;PREVIOUS OPERATOR TOKEN CODE
00A2                    PRVPRI  *=*+1           ;   "        "    PRIORITY (PRECEDENCE)
00A3                    CURPRI  *=*+1           ;CURRENT     "       "
00A4                    FPPTR   *=*+1           ;FPSTK POINTER (STARTS AT 0)
00A5                    OPPTR   *=*+1           ;OPSTK   "       "     "  "
00A6                    DISP    *=*+1           ;TEMP VAR FOR OPSTK PROCESSING LOOP, 1=>DISPLAY X WHEN DONE
00A7                    BITINT  *=*+1           ;1-32: NUMBER OF BITS IN OCTAL & HEX ARITHMETIC
                        ;DAYTMP                 ;TEMP VAR FOR DAYSUB
                        ;CHRIND                 ;INDEX INTO CHRTAB
                        ;COUNT                  ;TEMP COUNT VAR
                        ;LOP                    ;FOR SAND, SOR, SXOR  0=>AND, 1=>OR, 2=> XOR
                        ;SHFFLG                 ;FOR SRSHF,SLSHF  0=>LEFT, 1=>RIGHT
00A8                    TO      *=*+1           ;TEMP VAR (ALL OF ABOVE)
00A9                    NEGFLG  *=*+1           ;PL=> POSITIVE, MI=>NEGATIVE NUMBER
00AA                    INTFLG  *=*+1           ;0=>X & Y BOTH INTEGER IN ROOT, POWER
00AB                    NUMFLG  *=*+1           ;1=> PREVIOUS THING DISPLAYED WAS A NUMBER
00AC                    ASAVE   *=*+1           ;REG A SAVE LOC
00AD                    XSAVE   *=*+1
00AE                    YSAVE   *=*+1
00AF                    MEMNUM  *=*+1           ;MEMORY NUMBER
00B0                    PRVSTK  *=*+1           ;FOR DSPSTK: PREVIOUS ROWCRS VALUE AT END OF STACK

00B1                    BITBIN  *=*+4           ;2^(BITINT-1)-1
00B5                    BITBN2  *=*+4           ;(2^BITINT)-1
00B9                    BINMIN  *=*+4           ;-(2^(BITINT-1))  MSB-LSB = COMP(BITBIN)
00BD                    BINARY  *=*+4           ;FRO IN BINARY FORMAT
```

```
00C1                    BIN2    *=*+4           ;SECOND BINARY #

00C5                    QUADFLG *=*+1           ;SIN QUADRANT FLAG

00C6                    PC      *=*+2           ;PROGRAM COUNTER LSB,MSB (INIT TO PRGMEM)
0002                    EXEC    =       2
0001                    STOPRG  =       1
00C8                    PROG    *=*+1           ;0=>IMMEDIATE MODE, 1=>STORING PROG, 2=> EXECUTING
00C9                    TRACE   *=*+1           ;1 => TRACE ON (DISPLAY ALL PROGRAM EXECUTION)
00CA                    DSPFLG  *=*+1           ;1=> DO NOT DISPLAY OR PRINT ANYTHING (PROGRAM EXECUTING)
00CB                    SSTFLG  *=*+1           ;1=>DO SINGLE STEP (EXECUTE ONE INSTRUCTION)
00CC                    SSTOLD  *=*+1           ;SSTFLG FROM PREVIOUS LOOP

                        ;                               FP TO ASCII CONVERSION VARS
00CD                    SEFORM  *=*+1           ;1=>EFORM
00CE                    FIXNUM  *=*+1           ;FIX 0-9
00CF                    MANTLN  *=*+1           ;LENGTH OF MANTISSA
00D0                    SSIGN   *=*+1           ;BIT 8 IS SIGN BIT
00D1                    SMSD    *=*+1           ;SAVE MSD OF FRO  (FRO+1)
00D2                    CONFLG  *=*+1           ;CONVERSION MSG LSB
00D3                    SCONFG  *=*+1           ;SAVED CONFLG FROM PREVIOUS LOOP

                                *=FRO           ;FLOATING POINT RAM
```

COLLEEN CALCULATOR, BY C SHAW

```
                        *=$500
0500                TOKBUF  *=*+TOKLEN      ;TOKEN STRING BUFFER
                            *=LBPR1         ;FLOATING POINT
                        *=$600
0600                FPSTK   *=FPSLEN*FPREC+*        ;FLOATING POINT NUMBER STACK
                            *=*-1/256+1*256 ;GO TO NEXT PAGE BOUNDARY
0700                OPSTK   *=*+OPSLEN
                            *=*-1/256+1*256
0800                MEMORY  *=MEMLEN*FPREC+*
                            *=*-1/256+1*256
0B00                BLKBUF  *=*+LINLEN      ;ALL BLANKS
0B26                CTLRS   *=*+LINLEN      ;ALL CTRL R'S (HORIZ. LINES)
0B4C                MODFAC  *=*+FPREC       ;INT (Y/X) AFTER MOD
0B52                FTEMP   *=*+FPREC       ;MY OWN TEMP F.P. REG
0B58                FTEMP2  *=*+FPREC       ;ANOTHER TEMP FP REG
0B5E                FPX     *=*+FPREC       ;X REG SAVED DURING STORE PROGRAM MODE

0B64                LDCSAV  *=*+1           ;SAVE PART CHAR FOR LDCHAR
0B65                XSAVE2  *=*+1           ;SAVE X REG FOR DEGREE CONVERSION
0B66                YSAVE2  *=*+1           ;SAVE Y REG
0B67                DUEFLG  *=*+1           ;0 => ANNUITY DUE/FV, 1=>ORDINARY ANNUITY/FV
                    ;                        2 => ANNUITY DUE/PV, 3 => ORDINARY ANNUITY/PV
0B68                ANNFLG  *=*+1           ;0 => ANNUITY, 1=> COMPOUND INTEREST
0B69                ENTFLG  *=*+1           ;0 => ENTER VALUE, 1 => FIND VALUE (FOR INTEREST EQNS.)
0B6A                ITER    *=*+1           ;ITERATION COUNTER FOR FINDING INTEREST
0B6B                DMFLG   *=*+1           ;1=>DON'T DISPLAY MEM, EVEN IN MEM AREA
0B6C                ERRFLG  *=*+1           ;1=>ALREADY HAVE DISPLAYED ERROR MSG, DON'T DO MORE
0B6D                MEMFLG  *=*+1           ;0=>ADD (SIGMA PLUS), 1=>SUB (SIGMA MINUS)
0B6E                CALPTR  *=*+1           ;POINTER TO CALSTK
                            *=BLKBUF+$80    ;USE SECOND HALF OF PAGE
0B80                CALSTK  *=*+$80         ;SUBROUTINE CALL STACK  (128/2 = 64 CALLS DEEP)

                            *=*-1/256+1*256
0C00                PRGMEM  *=*+PRGLEN      ;USER PROGRAM MEMORY
```

```
                        *=$A000-6       ;TEMP CARTRIDGE B
9FFA 00 00 FF           .BYTE   0,0,$FF,0,0,0   ;NO CARTRIDGE
9FFD 00 00 00
                        *=$A000-$800
                ;
                ;           INITIALIZATION
                ;
                ;
                ;                                       CARTRIDGE COLD/WARM START LOC
9800            START
9800 A2 00              LDX     #0              ;CLEAR ZERO PAGE RAM ($80-$FF)
9802 8A                 TXA
9803            INIT2
9803 95 80              STA     ZROPG,X
9805 E8                 INX
9806 10 FB              BPL     INIT2

9808 A0 05              LDY     #TOKBUF/256             ;CLEAR NON-ZERO-PAGE RAM
980A A2 10              LDX     #PC1MAX+1
980C 20 C4 A3           JSR     RAMCLR
                ;
980F A9 01              LDA     #LMARG          ;SET UP MARGINS
9811 85 52              STA     LMARGN
9813 8D F0 02           STA     CRSINH          ;<>0 => INHIBIT CURSOR
9816 A9 26              LDA     #RMARG
9818 85 53              STA     RMARGN

                ;                               OPEN KEYBOARD, (SCREEN OPENED BY OS)
981A A2 10              LDX     #KIOCB
981C A9 03              LDA     #OPEN
981E 9D 42 03           STA     ICCOM,X
9821 A9 BE              LDA     #KBUFF
9823 9D 44 03           STA     ICBAL,X
9826 A9 B6              LDA     #KBUFF/256
9828 9D 45 03           STA     ICBAH,X
982B A9 04              LDA     #INPUT
982D 9D 4A 03           STA     ICAX1,X
9830 20 56 E4           JSR     CIOV
                ;                               CHECK FOR ERROR????

9833 A9 09              LDA     #9
9835 85 CE              STA     FIXNUM          ;INIT TO FIX 9

9837 A9 95              LDA     #LPAD
9839 20 C1 A2           JSR     PUSHOP          ;INIT OPERATOR STACK WITH LPAD ON BOTTOM
983C A9 05              LDA     #TOKBUF/$100
983E 85 83              STA     TOKPTR+1

9840 20 39 AC           JSR     SCLPRO          ;INITIALIZE PC TO START OF PRGMEM AND CLEAR PRGMEM TO ALL STP'S

9843 A2 25              LDX     #LINLEN-1       ;INIT BLKBUF & CTLRS
9845            INIT4
9845 A9 20              LDA     #'
9847 9D 00 0B           STA     BLKBUF,X
984A A9 12              LDA     #'R-64
984C 9D 26 0B           STA     CTLRS,X
984F CA                 DEX
9850 10 F3              BPL     INIT4

9852 20 16 A6           JSR     SALG            ;DEFAULT IS ALGEBRAIC WITH OPERATOR PRECEDENCE
```

```
9852 20 16 A6                   JSR     SALG            ;DEFAULT IS ALGEBRAIC WITH OPERATOR PRECEDENCE
```

COLLEEN CALCULATOR, BY C SHAW

```
9855 A9 10                      LDA     #16
9857 20 7D A6                   JSR     SBITS2

                        ;                               INIT SCREEN DISPLAY
                        ;                               ;LINE 0-1    "ALG  RAD  . . .
985A A9 FD                      LDA     #STATLN
985C A0 B8                      LDY     #STATLN/256
985E 20 4F 9C                   JSR     SETMSG
9861 20 5E A3                   JSR     PUTCHS

9864 A2 00                      LDX     #0
9866 20 31 A3                   JSR     PTLIN1          ;LINE 2

                        ;                               ;LINE 3   "¦ STACK     ¦REG . . .
9869 A9 37                      LDA     #STKLIN
986B A0 B9                      LDY     #STKLIN/256
986D 20 4F 9C                   JSR     SETMSG          ;SET UP MESSAGE IN TOKBUF
9870 20 5E A3                   JSR     PUTCHS

9873 A2 03                      LDX     #3              ;LINE 4
9875 20 31 A3                   JSR     PTLIN1

9878 A2 00                      LDX     #0              ;LINES 5-14   "¦X       ¦0       0¦"
987A                    PTLP
987A 86 A8                      STX     T0
987C A9 7C                      LDA     #'¦
987E 20 F2 A2                   JSR     PTCHR
9881 A6 A8                      LDX     T0
9883 BD D5 B6                   LDA     CHTAB2,X        ;X, Y, Z, T, OR BLANK
9886 20 F2 A2                   JSR     PTCHR
9889 20 58 A3                   JSR     BLNK16
988C A9 7C                      LDA     #'¦
988E 20 F2 A2                   JSR     PTCHR
9891 A5 A8                      LDA     T0
9893 18                         CLC
9894 69 30                      ADC     #'0
9896 20 F2 A2                   JSR     PTCHR           ;'0 - '9
9899 A2 11                      LDX     #17
989B 20 5A A3                   JSR     BLNKS
989E A9 7C                      LDA     #'¦
98A0 20 F2 A2                   JSR     PTCHR
98A3 A6 A8                      LDX     T0
98A5 E8                         INX
98A6 E0 0A                      CPX     #10
98A8 D0 D0                      BNE     PTLP

98AA A2 06                      LDX     #6              ;LINE 15
98AC 20 31 A3                   JSR     PTLIN1

98AF 20 51 9D                   JSR     DSPSTK          ;DISPLAY STACK (ONLY HAVE X=0)
98B2 20 1B A7                   JSR     DMCLR           ;DISPLAY 0'S IN MEM AREA
98B5 20 5D 9E                   JSR     FDSCOM          ;DISPLAY X = 0 IN SCROLL AREA

98B8 A9 01                      LDA     #1
98BA 85 A0                      STA     INTERM          ;DISPLAY INTERMEDIATE RESULTS IN ALGP
```

```
98BC                LOOP
98BC AD F0 02               LDA     CRSINH          ;BREAK KEY HIT?
98BF C9 00                  CMP     #$0
98C1 D0 11                  BNE     MAIN02          ;NO.

98C3 EE F0 02               INC     CRSINH          ;INHIBIT CURSOR
98C6 A9 80                  LDA     #$80            ;YES. CLEAR FLAG
98C8 85 11                  STA     BRKKEY
98CA A5 C8                  LDA     PROG            ;STOP PROGRAM EXECUTION
98CC 29 01                  AND     #1
98CE 20 24 AD               JSR     SEND            ;DISPLAY STACK, CHANGE PROG
98D1 20 73 9E               JSR     FDSP0           ;DISPLAY CURRENT X IN SCROLL AREA
98D4                MAIN02
98D4 A5 C8                  LDA     PROG
98D6 C9 01                  CMP     #STOPRG         ;STORE PROGRAM?
98D8 F0 03                  BEQ     MAIN03          ;NO
98DA 4C 4C 99               JMP     NOSTOR

                    ;                                       STORE PROGRAM MODE
98DD                MAIN03
98DD 20 97 9D               JSR     DSPRG           ;DISPLAY OLD VALUE IN PROGRAM LOC

98E0 20 79 9A               JSR     LEX             ;GET NEXT TOKEN FROM PROGRAM MEM
98E3 20 D2 A2               JSR     PUTDEL

98E6 A5 81                  LDA     TOKCOD          ;CHECK FOR SPECIAL COMMAND
98E8 A2 06                  LDX     #SPCLEN
98EA                LOOP3
98EA DD B4 B6               CMP     SPCTBL,X
98ED D0 1A                  BNE     LOOP4

98EF A9 00                  LDA     #0              ;SPECIAL COMMAND FOUND
98F1 85 AB                  STA     NUMFLG          ;ALWAYS ON SEPARATE LINE
98F3 A9 16                  LDA     #22
98F5 85 54                  STA     ROWCRS
98F7 85 55                  STA     COLCRS
98F9 A4 82                  LDY     TOKPTR
98FB 20 F6 9D               JSR     PUTCMD          ;DISPLAY COMMAND
98FE 20 85 A3               JSR     PUTCRP
9901 A5 81                  LDA     TOKCOD
9903 20 C5 A0               JSR     SUBCAL          ;CALL SUBROUTINE
9906 4C BC 98               JMP     LOOP            ;CONTINUE

9909                LOOP4
9909 CA                     DEX
990A 10 DE                  BPL     LOOP3           ;TRY NEXT ONE

990C C9 96                  CMP     #NUMBER         ;NOT SPECIAL COMMAND => SAVE        NUMBER?
990E D0 24                  BNE     STPR40          ;NO.
9910 20 62 A2               JSR     PCNCHK          ;CHECK PC TO SEE IF ROOM FOR NUMBER
9913 B0 A7                  BCS     LOOP            ;ERROR   END OF MEM

9915 20 A7 DD               JSR     FSTOR           ;STORE FRO IN PRGMEM
9918 A0 07                  LDY     #FPREC+1
991A A9 96                  LDA     #NUMBER
991C 91 C6                  STA     (PC),Y
991E A0 00                  LDY     #0
9920 91 C6                  STA     (PC),Y
9922 20 7A A2               JSR     PCADDN          ;MOVE PC PAST NUMBER
```

```
9922 20 7A A2                  JSR     PCADDN          ;MOVE PC PAST NUMBER
```

COLLEEN CALCULATOR, BY C SHAW

```
9925 20 A0 9E                  JSR     TOKNUM
9928 A9 16                     LDA     #22
992A 85 54                     STA     ROWCRS
992C 85 55                     STA     COLCRS
992E 20 07 9E                  JSR     DG70            ;PTTXTP TOKBUF
9931 4C 46 99                  JMP     STPR51

9934                   STPR40                          ;NOT A NUMBER
9934 A0 00                     LDY     #0
9936 91 C6                     STA     (PC),Y
9938 20 76 A2                  JSR     PCINC

                       ;                               ;DISPLAY NEW TOKEN AFTER OLD
993B                   STPR50
993B A9 16                     LDA     #22
993D 85 54                     STA     ROWCRS
993F 85 55                     STA     COLCRS
9941 A4 82                     LDY     TOKPTR
9943 20 F6 9D                  JSR     PUTCMD
9946                   STPR51
9946 20 85 A3                  JSR     PUTCRP          ;PUT CR ON PRINTER (IF ON)

9949 4C BC 98                  JMP     LOOP
```

```
994C                    NOSTOR                          ;NOT STORE PROGRAM MODE
994C 20 51 9D                   JSR     DSPSTK          ;DISPLAY STACK

994F A5 D2                      LDA     CONFLG          ;SAVE CONVERSION INDICATOR
9951 85 D3                      STA     SCONFG

9953 20 79 9A                   JSR     LEX
9956 90 03                      BCC     NOST02
9958 4C BC 98                   JMP     LOOP            ;EXEC ERROR (OUT OF EXEC MODE)
995B                    NOST02

995B 20 D2 A2                   JSR     PUTDEL

995E A9 00                      LDA     #0
9960 85 9F                      STA     NOPFLG
9962 A6 CB                      LDX     SSTFLG
9964 86 CC                      STX     SSTOLD
9966 85 CB                      STA     SSTFLG
9968 A5 81                      LDA     TOKCOD
996A C9 96                      CMP     #NUMBER         ;NUMBER?
996C D0 13                      BNE     MAIN05          ;NO.  SKIP

996E A5 99                      LDA     RPNALG          ;RPN?
9970 D0 06                      BNE     MAIN04          ;NO.
9972 20 4E A0                   JSR     FLD1T
9975 20 AA A0                   JSR     FPUSH1          ;YES.  SAVE PREVIOUS # ON STACK
9978                    MAIN04
9978 20 D2 A2                   JSR     PUTDEL
997B 20 73 9E                   JSR     FDSP0           ;CONVERT BACK TO ASCII AND DISPLAY
997E 4C 48 9A                   JMP     ENDLP

9981                    MAIN05                          ;NOT NUMBER
9981 20 EE 9C                   JSR     DSPCOM          ;DISPLAY COMMAND
9984 A0 00                      LDY     #0
9986 84 AB                      STY     NUMFLG

9988 A5 81                      LDA     TOKCOD
998A 20 D2 A1                   JSR     GETPRI
998D 85 A3                      STA     CURPRI
998F C9 0D                      CMP     #PHIGH
9991 90 11                      BCC     MAIN40

9993 A5 81                      LDA     TOKCOD          ;SPECIAL OR HIGH
9995 20 C5 A0                   JSR     SUBCAL          ;EXECUTE SUBROUTINE
9998 A5 A3                      LDA     CURPRI
999A C9 0D                      CMP     #PHIGH
999C D0 03                      BNE     MAIN35
999E 20 5D 9E                   JSR     FDSCOM          ;HIGH
99A1                    MAIN35
99A1 4C 48 9A                   JMP     ENDLP

99A4                    MAIN40
99A4 A6 99                      LDX     RPNALG
99A6 D0 17                      BNE     MAIN60
99A8 C9 03                      CMP     #PLRPAR+1       ;RPN
99AA B0 08                      BCS     MAIN50
99AC A9 0D                      LDA     #KEYMSG
99AE 20 E5 9B                   JSR     ERRSUB          ;( ) = NOT LEGAL IN RPN
99B1 4C 4C 9A                   JMP     ENDLP3
```

```
99B4                    MAIN50
99B4 A5 81                      LDA     TOKCOD
99B6 20 C5 A0                   JSR     SUBCAL          ;EXECUTE SUBROUTINE
99B9 20 5D 9E                   JSR     FDSCOM
99BC 4C 4C 9A                   JMP     ENDLP3

99BF                    MAIN60                          ;NOT RPN
                        ;                               CHECK FOR 2 OPS IN A ROW
99BF A6 81                      LDX     TOKCOD
99C1 E0 92                      CPX     #LPAR           ;OP CAN BE FOLLOWED BY '('
99C3 F0 15                      BEQ     MAIN70
99C5 C9 03                      CMP     #PLRPAR+1
99C7 90 02                      BCC     MAIN65          ;')' AND '=' CAN BE FOLLOWED BY OP
99C9 E6 9F                      INC     NOPFLG          ;2-VAR OPERATOR
99CB                    MAIN65
99CB A6 9E                      LDX     OPFLG
99CD F0 0B                      BEQ     MAIN70
99CF A9 00                      LDA     #TOPMSG         ;2 OPS IN A ROW IS ILLEGAL: THIS TIME IGNORE 1ST OP
99D1 20 E5 9B                   JSR     ERRSUB
99D4 20 B0 A2                   JSR     POPOP           ;DISCARD PREV OP
99D7 20 75 A0                   JSR     FPOP1           ;DISCARD EXTRA #

99DA                    MAIN70
99DA A5 81                      LDA     TOKCOD


99DC                    MAIN76
99DC C9 92                      CMP     #LPAR
99DE D0 08                      BNE     MAIN80
99E0 20 C1 A2                   JSR     PUSHOP          ;PUSH LPAR
99E3 E6 9F                      INC     NOPFLG
99E5 4C 48 9A                   JMP     ENDLP
99E8                    MAIN80
99E8 A9 00                      LDA     #0
99EA 85 A6                      STA     DISP

99EC                    WLOOP
99EC 20 B0 A2                   JSR     POPOP
99EF 85 A1                      STA     PREVOP
99F1 20 D2 A1                   JSR     GETPRI
99F4 85 A2                      STA     PRVPRI

99F6 A5 81                      LDA     TOKCOD
99F8 C9 93                      CMP     #RPAR
99FA D0 19                      BNE     WLP10
99FC A5 A1                      LDA     PREVOP          ;TOKCOD =RPAR
99FE C9 92                      CMP     #LPAR
9A00 F0 3F                      BEQ     ENDWLP          ;PREVOP=LPAR
9A02 C9 95                      CMP     #LPAD
9A04 D0 06                      BNE     WLP05
9A06 20 C1 A2                   JSR     PUSHOP
9A09 4C 41 9A                   JMP     ENDWLP
9A0C                    WLP05
9A0C 20 C5 A0                   JSR     SUBCAL          ;EXECUTE SUBROUTINE
9A0F 20 F0 A1                   JSR     INTDSP
9A12 4C EC 99                   JMP     WLOOP

9A15                    WLP10
9A15 C9 94                      CMP     #EQUAL
9A17 D0 06                      BNE     WLP20
```

```
9A19 A5 A1                  LDA     PREVOP
9A1B C9 92                  CMP     #LPAR
9A1D F0 CD                  BEQ     WLOOP
9A1F                WLP20                       ;NOT (TOKCOD = EQUAL AND PREVOP = LPAR)
9A1F A5 A2                  LDA     PRVPRI
9A21 C5 A3                  CMP     CURPRI
9A23 90 0B                  BCC     WLP30
9A25 A5 A1                  LDA     PREVOP      ;PRVPRI>=CURPRI
9A27 20 C5 A0               JSR     SUBCAL      ;EXECUTE SUBROUTINE
9A2A 20 F0 A1               JSR     INTDSP
9A2D 4C EC 99               JMP     WLOOP

9A30                WLP30                       ;PRVPRI<CURPRI
9A30 A5 A1                  LDA     PREVOP
9A32 20 C1 A2               JSR     PUSHOP
9A35 A5 81                  LDA     TOKCOD
9A37 C9 94                  CMP     #EQUAL
9A39 F0 06                  BEQ     ENDWLP
9A3B 20 C1 A2               JSR     PUSHOP
9A3E 20 9C A0               JSR     FPUSH0


9A41                ENDWLP
9A41 A5 A6                  LDA     DISP
9A43 F0 03                  BEQ     ENDLP
9A45 20 5D 9E               JSR     FDSCOM

9A48                ENDLP
9A48 A5 9F                  LDA     NOPFLG
9A4A 85 9E                  STA     OPFLG
9A4C                ENDLP3
9A4C A5 CC                  LDA     SSTOLD      ;SINGLE STEP?
9A4E F0 06                  BEQ     ENDLP4      ;NO.
9A50 A9 00                  LDA     #0          ;YES.  GO BACK TO IMMEDIATE MODE
9A52 85 C8                  STA     PROG
9A54 85 CB                  STA     SSTFLG
9A56                ENDLP4
9A56 A5 D2                  LDA     CONFLG
9A58 38                     SEC
9A59 E5 D3                  SBC     SCONFG      ;CONFLG SAME ON THIS & PREVIOUS LOOP (SET OR CLEAR)?
9A5B D0 02                  BNE     EDSKP2      ;NO.  NO CHANGE NOW
9A5D 85 D2                  STA     CONFLG      ;YES.  CLEAR (NO LONGER DOING CONVERSION)
9A5F                EDSKP2
9A5F 4C BC 98               JMP     LOOP

                    ;                           END OF MAIN PROGRAM LOOP
```

```
                        ;                                   LEXICAL ANALYZER
                        ;
                        ;                       FETCH NEXT TOKEN FROM TERMINAL AND
                        ;                       SET UP TOKEN CODE IN TOKCOD, PUT STRING IN TOKBUF
                        ;
9A62                    LXINIT                              ;SUBROUTINE TO DISPLAY '>', SET UP CURSOR
9A62 A9 00                      LDA     #0
9A64 8D F0 02                   STA     CRSINH              ;CURSOR ON
9A67 A9 02                      LDA     #LMARG+1
9A69 85 55                      STA     COLCRS
9A6B A9 17                      LDA     #23
9A6D 85 54                      STA     ROWCRS              ;START AT BOTTOM OF SCREEN
9A6F A9 3E                      LDA     #'>
9A71 20 F2 A2                   JSR     PTCHR               ;TEST CHAR

9A74 A9 00                      LDA     #0
9A76 85 82                      STA     TOKPTR
9A78 60                         RTS


9A79                    LEX
9A79 A9 00                      LDA     #0                  ;CLEAR FLAG => NO ERROR THIS TIME
9A7B 8D 6C 0B                   STA     ERRFLG

9A7E A5 C8                      LDA     PROG
9A80 C9 02                      CMP     #EXEC               ;EXECUTING PROGRAM?
9A82 D0 11                      BNE     NOEXEC              ;NO.
9A84 20 3C A2                   JSR     NCHKLD              ;YES.  LOAD TOKEN AND CHECK FOR NUMBER
9A87 B0 0B                      BCS     EXEC20              ;ERROR
9A89 D0 03                      BNE     EXEC10              ;NOT NUMBER
9A8B 4C 7A A2                   JMP     PCADDN              ;NUMBER         MOVE PC PAST #
9A8E                    EXEC10
9A8E 20 76 A2                   JSR     PCINC               ;NOT NUMBER
9A91 20 61 A4                   JSR     UNPKEY              ;UNPACK KEYWORD
9A94                    EXEC20
9A94 60                         RTS



9A95                    NOEXEC
9A95 20 62 9A                   JSR     LXINIT              ;DISPLAY '>' ; INIT CURSOR
9A98 20 E8 A0                   JSR     GTCHR               ;A=NEXT CHAR
9A9B C9 20                      CMP     #'
9A9D F0 DA                      BEQ     LEX
9A9F C9 9C                      CMP     #DELLIN
9AA1 F0 D6                      BEQ     LEX

                        ;
                        ;                                   CHECK FOR SINGLE CHAR TOKENS
                        ;
9AA3 A2 0D                      LDX     #TOKCLN
9AA5                    LXLP20
9AA5 DD 9C B6                   CMP     TOKCHR,X
9AA8 D0 0B                      BNE     LEX30

9AAA BD AA B6                   LDA     TOKTBL,X
9AAD 85 81                      STA     TOKCOD
9AAF 20 61 A4                   JSR     UNPKEY              ;UNPACK KEYWORD IN CASE POWER, PERCENT, SSTEP, BSTEP, ETC
9AB2 4C DE 9B                   JMP     LXRTN2
```

```
9AB5                        LEX30
9AB5 CA                             DEX
9AB6 10 ED                          BPL     LXLP20
```

```
                        ;
                        ;                                   CHECK FOR KEYWORD (ALPHA)
                        ;
9AB8 C9 41                      CMP     #'A
9ABA B0 03                      BCS     LEX05
9ABC                LEX02
9ABC 4C 41 9B                   JMP     LXNMCK
9ABF                LEX05
9ABF C9 5B                      CMP     #'Z+1
9AC1 B0 F9                      BCS     LEX02

9AC3 20 FC A3                   JSR     UNPINT
9AC6                KEYLP1
9AC6 20 0F A4                   JSR     UNPNUM

9AC9 F0 48                      BEQ     ENDLST          ;END OF LIST IF 0 COUNT
9ACB                KEYLP2
9ACB 20 FC A1                   JSR     LDCHR

9ACE A6 88                      LDX     KEYCHR
9AD0 DD 00 05                   CMP     TOKBUF,X
9AD3 F0 04                      BEQ     KEY06           ;MATCH
9AD5 90 30                      BCC     KEY20           ;NO MATCH:  HAVEN'T GONE FAR ENOUGH
9AD7 B0 3E                      BCS     KEYERR          ;NO MATCH:  HAVE GONE TOO FAR - GIVE ERROR MSG
9AD9                KEY06
9AD9 E8                         INX
9ADA 86 88                      STX     KEYCHR

9ADC E4 82                      CPX     TOKPTR          ;NOT END OF WORD
9ADE 90 EB                      BCC     KEYLP2
9AE0 F0 E9                      BEQ     KEYLP2

9AE2 84 8B                      STY     LDNBSV          ;NEED TO FETCH MORE CHARS FROM TERMINAL
9AE4 E6 82                      INC     TOKPTR          ;SAVE PREVIOUS CHAR
9AE6 20 E8 A0                   JSR     GTCHR

9AE9 C9 9C                      CMP     #DELLIN
9AEB D0 03                      BNE     KEY07
9AED 4C 79 9A                   JMP     LEX
9AF0                KEY07

9AF0 A4 8B                      LDY     LDNBSV
9AF2 C9 41                      CMP     #'A
9AF4 90 0C                      BCC     KEY10           ;END OF CHAR STRING (INCOMPLETE KEYWORD MATCH)
9AF6 C9 5B                      CMP     #'Z+1
9AF8 B0 08                      BCS     KEY10
9AFA A6 88                      LDX     KEYCHR
9AFC E4 8A                      CPX     KEYLN2
9AFE B0 07                      BCS     KEY20           ;AT END OF WORD => SAVE MATCH AND CONTINUE
9B00 90 C9                      BCC     KEYLP2          ;NOT END OF WORD => CONTINUE

9B02                KEY10
9B02 A5 8F                      LDA     KEYCNT
9B04 4C 27 9B                   JMP     ENDL20

9B07                KEY20
9B07 20 EA A3                   JSR     SAVMAT          ;SAVE LONGEST MATCH
                                                        ;TRY NEXT WORD IN LIST
9B0A 20 1B A4                   JSR     UNPNXT
```

```
9B0D 4C C6 9A                JMP     KEYLP1          ;CONTINUE


9B10                 KEY70
9B10 20 EA A3                JSR     SAVMAT          ;WORD NOT IN LIST => TAKE LONGEST MATCH
9B13                 ENDLST
9B13 A5 94                   LDA     KMATCH
9B15 D0 05                   BNE     ENDSKP
9B17                 KEYERR
9B17 A9 0D                   LDA     #KEYMSG
9B19 4C C8 9B                JMP     LEXERR          ;FIRST CHAR NOT IN KEYWORD
9B1C                 ENDSKP

9B1C A2 03                   LDX     #3              ;USE LONGEST MATCH
9B1E                 ENDL10
9B1E B5 90                   LDA     KYPTSV,X
9B20 95 8C                   STA     PKPTR,X
9B22 CA                      DEX
9B23 10 F9                   BPL     ENDL10
9B25 A5 8F                   LDA     KEYCNT

9B27                 ENDL20
9B27 85 81                   STA     TOKCOD
9B29 20 DF A3                JSR     SAVCHR
9B2C A5 8E                   LDA     KYLFRT
9B2E 85 80                   STA     LFRT
9B30 A2 00                   LDX     #TOKBUF         ;UNPACK ENTIRE KEYWORD INTO TOKBUF
9B32 20 3F A4                JSR     UNPACK
9B35 84 82                   STY     TOKPTR
9B37 4C D8 9B                JMP     LEXRTN

9B3A                 ENDWRD
9B3A E6 82                   INC     TOKPTR          ;SAVE LAST CHAR
9B3C A5 8F                   LDA     KEYCNT
9B3E 4C D6 9B                JMP     LXSAV0
```

```
                        ;
                        ;                                   CHECK FOR NUMBER
                        ;
9B41                    LXNMCK
9B41 C9 2E                      CMP     #'.
9B43 D0 0F                      BNE     LXNDOT

9B45 A6 87                      LDX     DHOFLG          ;DEC?
9B47 D0 10                      BNE     LXERR2          ;NO.  '.' NOT ALLOWED
9B49 E6 82                      INC     TOKPTR          ;SAVE "."
9B4B 20 60 A1                   JSR     GETDHO          ;GET DEC, HEX OR OCT DIGIT (ACCORDING TO DHOFLG)
9B4E F0 22                      BEQ     LXHVDT
9B50 A9 0D                      LDA     #KEYMSG         ;'.' WITH NO DIGITS  "NOT VALID COMMAND"
9B52 D0 74                      BNE     LEXERR          ;JMP

9B54                    LXNDOT
9B54 20 6C A1                   JSR     DHOCHK
9B57 F0 04                      BEQ     LXLP40
9B59                    LXERR2
9B59 A9 17                      LDA     #NOMMSG
9B5B D0 6B                      BNE     LEXERR          ;NO MATCH AT ALL SO ILLEGAL CHAR
                        ;                               HAVE NUMBER
9B5D                    LXLP40
9B5D A6 82                      LDX     TOKPTR
9B5F E0 11                      CPX     #NUMLEN+1       ;LIMIT TO NUMLEN CHARS
9B61 B0 63                      BCS     LENERR
9B63 20 60 A1                   JSR     GETDHO
9B66 F0 F5                      BEQ     LXLP40          ;KEEP GETTING DIGITS

9B68 C9 2E                      CMP     #'.
9B6A D0 11                      BNE     LXND2
9B6C A6 87                      LDX     DHOFLG          ;DEC?
9B6E D0 E9                      BNE     LXERR2          ;NO.  '.' NOT ALLOWED

9B70 E6 82                      INC     TOKPTR          ;SAVE "."
9B72                    LXHVDT
9B72 A6 82                      LDX     TOKPTR          ;CHECK FOR BUFFER OVERFLOW
9B74 E0 11                      CPX     #NUMLEN+1       ;LIMIT TO NUMLEN CHARS
9B76 B0 4E                      BCS     LENERR          ;TOO MANY DIGITS
9B78 20 60 A1                   JSR     GETDHO
9B7B F0 F5                      BEQ     LXHVDT

9B7D                    LXND2
9B7D A6 87                      LDX     DHOFLG
9B7F D0 30                      BNE     LXNUM           ;OCTAL OR HEX => NO EXP ALLOWED

9B81 C9 45                      CMP     #'E             ;DECIMAL   CHECK FOR EXPONENT
9B83 D0 2C                      BNE     LXNUM           ;NO E=> END OF NUMBER

9B85 E6 82                      INC     TOKPTR          ;SAVE E
9B87 20 60 A1                   JSR     GETDHO
9B8A F0 1B                      BEQ     LXGT2           ;HAVE DIGIT

9B8C C9 2B                      CMP     #'+             ;NOT A DIGIT.  '+' OR '-'?
9B8E F0 04                      BEQ     LX50
9B90 C9 2D                      CMP     #'-
9B92 D0 0B                      BNE     LX60            ;E IS NOT FOR EXPONENT=> DONE WITH NUMBER
9B94                    LX50
9B94 E6 82                      INC     TOKPTR          ;E IS FOR EXPONENT    => SAVE '+' OR '-'
```

```
9B96 20 60 A1              JSR     GETDHO
9B99 F0 0C                 BEQ     LXGT2           ;HAVE DIGIT
9B9B A9 2A                 LDA     #NODMSG         ;NO DIGIT AFTER + OR - IN EXPONENT
9B9D D0 29                 BNE     LEXERR          ;JMP

9B9F                LX60
9B9F 20 DF A3              JSR     SAVCHR          ;SAVE E FOR NEXT TOKEN
9BA2 C6 82                 DEC     TOKPTR
9BA4 4C B1 9B              JMP     LXNUM

9BA7                LXGT2
9BA7 20 60 A1              JSR     GETDHO          ;GET 2ND DIGIT OF EXPONENT
9BAA D0 05                 BNE     LXNUM           ;NO 2ND DIGIT
9BAC 20 60 A1              JSR     GETDHO          ;HAVE 2ND DIGIT. IS THERE 3RD?
9BAF F0 15                 BEQ     LENERR          ;ERROR - EXPONENT TOO LARGE

9BB1                LXNUM
9BB1 20 DF A3              JSR     SAVCHR          ;SAVE LAST CHAR FOR NEXT TOKEN
9BB4                LXN2
9BB4 A0 00                 LDY     #0
9BB6 A9 9B                 LDA     #CR
9BB8 91 82                 STA     (TOKPTR),Y
9BBA 20 55 A0              JSR     FSTOT
9BBD 20 43 A5              JSR     SNUMB           ;ASCII -> FP
9BC0 A9 96                 LDA     #NUMBER
9BC2 85 81                 STA     TOKCOD
9BC4 D0 18                 BNE     LXRTN2          ;JMP

9BC6                LENERR
9BC6 A9 82                 LDA     #DIGMSG         ;TOO MANY DIGITS
9BC8                LEXERR
9BC8 20 E5 9B              JSR     ERRSUB
9BCB 4C 79 9A              JMP     LEX             ;TRY AGAIN

9BCE                LXSAV1
9BCE 85 81                 STA     TOKCOD          ;SAVE TOKEN CODE
9BD0 20 DF A3              JSR     SAVCHR          ;SAVE LAST CHAR FOR NEXT TOKEN
9BD3 4C D8 9B              JMP     LEXRTN

9BD6                LXSAV0
9BD6 85 81                 STA     TOKCOD          ;DON'T NEED TO SAVE ANY CHARS E.G. '('
9BD8                LEXRTN
9BD8 A0 00                 LDY     #0
9BDA A9 9B                 LDA     #CR
9BDC 91 82                 STA     (TOKPTR),Y
9BDE                LXRTN2
9BDE A0 01                 LDY     #1
9BE0 8C F0 02              STY     CRSINH          ;CURSOR OFF
9BE3 18                    CLC                     ;NO ERROR
9BE4                INIT                                ;POWER UP INIT: JUST RETURN
9BE4 60                    RTS
                    ;                                                   END OF LEX
```

```
9BE5                ERRSUB
                    ;                                   OUTPUT "ERROR - ";MESSAGE      A=LSB OF ADDRESS
9BE5 AE 6C 0B               LDX     ERRFLG
9BE8 D0 FA                  BNE     INIT                ;RETURN IF ERROR ALREADY DISPLAYED
9BEA EE 6C 0B               INC     ERRFLG              ;<- 1  SET FLAG
9BED AA                     TAX
9BEE A5 54                  LDA     ROWCRS
9BF0 48                     PHA                         ;SAVE OLD CURSOR LOC SO IT CAN BE RESTORED LATER
9BF1 A5 55                  LDA     COLCRS
9BF3 48                     PHA
9BF4 8A                     TXA
9BF5 48                     PHA

9BF6 20 13 9C               JSR     ERRSB2              ;DO SOUND, SET UP TO OUTPUT "ERROR - "
9BF9 20 BA A3               JSR     PTTXTP              ;PUT ON SCREEN AND PRINTER
9BFC A9 02                  LDA     #2
9BFE 85 55                  STA     COLCRS              ;ONE SPACE
9C00 68                     PLA                         ;RELOAD MESSAGE ADDR
9C01 A0 B7                  LDY     #ERRTBL/256
9C03 20 4F 9C               JSR     SETMSG              ;SET UP MESSAGE IN TOKBUF
9C06 20 BA A3               JSR     PTTXTP              ;PUT TEXT ON SCREEN AND PRINTER

9C09 68                     PLA
9C0A 85 55                  STA     COLCRS
9C0C 68                     PLA
9C0D 85 54                  STA     ROWCRS
9C0F 38                     SEC
9C10 A2 01                  LDX     #1                  ;NE => ERROR
9C12 60                     RTS


9C13                ERRSB2
9C13 A9 00                  LDA     #0
9C15 85 CA                  STA     DSPFLG              ;TURN DISPLAY ON
9C17 A5 C8                  LDA     PROG
9C19 29 01                  AND     #1
9C1B 85 C8                  STA     PROG                ;STOP EXECUTION, IF ANY

9C1D A9 40                  LDA     #$40                ;OUTPUT ERROR SOUND
9C1F 8D 00 D2               STA     AUDF1
9C22 A9 AF                  LDA     #$AF
9C24 8D 01 D2               STA     AUDC1
9C27 A0 80                  LDY     #$80                ;DELAY
9C29 A2 01                  LDX     #1
9C2B 8E F0 02               STX     CRSINH              ;CURSOR OFF
9C2E CA                     DEX                         ;0
9C2F 86 AB                  STX     NUMFLG              ;LAST THING OUTPUT WILL NOT BE A NUMBER
9C31                SNDLP1
9C31 CA                     DEX
9C32 D0 FD                  BNE     SNDLP1
9C34 88                     DEY
9C35 D0 FA                  BNE     SNDLP1
9C37 8C 01 D2               STY     AUDC1               ;TURN SOUND OFF

9C3A 20 81 A3               JSR     PTCRPN              ;PUT CR ON PRINTER IF PREVIOUS WAS NUMBER

                    ;                                   PUT "ERROR"
9C3D A9 17                  LDA     #23
```

```
9C3F 85 54              STA     ROWCRS
9C41 20 DE A2           JSR     PUTCR
9C44 A9 02              LDA     #2
9C46 85 55              STA     COLCRS          ;ONE SPACE
9C48 A2 07              LDX     #7
9C4A A9 C1              LDA     #ERRMSG
9C4C A0 B6              LDY     #ERRMSG/256
9C4E 60                 RTS


9C4F                SETMSG                      ;SET UP MESSAGE IN TOKBUF: A=MSG LSB, Y=MSG MSB
9C4F A2 00              LDX     #TOKBUF
9C51 86 82              STX     TOKPTR
9C53 85 8C              STA     PKPTR
9C55 84 8D              STY     PKPTR+1
9C57 A0 00              LDY     #0
9C59 84 80              STY     LFRT
9C5B B1 8C              LDA     (PKPTR),Y
9C5D 85 8A              STA     KEYLN2
9C5F 98                 TYA
9C60 20 3B A4           JSR     UNPCK2
9C63 98                 TYA
9C64 AA                 TAX                     ;# OF CHARS

9C65 A9 00              LDA     #TOKBUF
9C67 A0 05              LDY     #TOKBUF/256
9C69 60                 RTS


9C6A                PUTMSG                      ;PUT MESSAGE ON BOTTOM LINE OF SCREEN & PRINTER
                    ;                           INPUT: A=MSG LSBYTE
9C6A A0 B8              LDY     #PROMSG/256     ;MSG MSB
9C6C                PTMSG2                      ;INPUT A=MSG LSB, Y=MSG MSB
9C6C A2 02              LDX     #2
9C6E 86 55              STX     COLCRS
9C70 A2 17              LDX     #23
9C72 86 54              STX     ROWCRS
9C74 20 4F 9C           JSR     SETMSG          ;SET UP MSG IN TOKBUF
9C77 4C BA A3           JMP     PTTXTP          ;PUT TOKBUF ON SCREEN & PRINTER
```

```
9C7A                    BINCHK                          ;CHECK TO SEE IF BINARY IS WITHIN RANGE
9C7A A5 BD                      LDA     BINARY
9C7C 30 12                      BMI     BINC10
9C7E A2 00                      LDX     #0              ;POSITIVE
9C80                    BINLP1
9C80 B5 BD                      LDA     BINARY,X
9C82 D5 B1                      CMP     BITBIN,X
9C84 F0 04                      BEQ     BINOK
9C86 A9 36                      LDA     #BOMSG
9C88 B0 16                      BCS     BINERR          ;TOO LARGE
9C8A                    BINOK
9C8A E8                         INX
9C8B E0 04                      CPX     #4
9C8D D0 F1                      BNE     BINLP1
9C8F 60                         RTS                     ;OK
9C90                    BINC10
9C90 A2 00                      LDX     #0
9C92                    BINLP2
9C92 B5 BD                      LDA     BINARY,X
9C94 D5 B9                      CMP     BINMIN,X
9C96 90 06                      BCC     BINER2          ;TOO SMALL
9C98 E8                         INX
9C99 E0 04                      CPX     #4
9C9B D0 F5                      BNE     BINLP2
9C9D 60                         RTS                     ;OK
9C9E                    BINER2
9C9E A9 36                      LDA     #BOMSG          ;BINARY UNDERFLOW
9CA0                    BINERR
9CA0 4C E5 9B                   JMP     ERRSUB

9CA3                    BINFP                           ;4 BYTE BINARY TO FP
9CA3 20 7A 9C                   JSR     BINCHK          ;LIMIT TO BITINT BITS
9CA6 A5 BD                      LDA     BINARY
9CA8                    BINFP2
9CA8 85 A9                      STA     NEGFLG
9CAA 10 03                      BPL     BIN10
9CAC 20 1E A5                   JSR     S2CMP           ;TAKE ABSOLUTE VALUE
9CAF                    BIN10
9CAF A5 BD                      LDA     BINARY          ;65536*IFP(BINARY+1,BINARY)+IFP(BINARY+3,BINARY+2)
9CB1 85 D5                      STA     FR0+1
9CB3 A5 BE                      LDA     BINARY+1
9CB5 85 D4                      STA     FR0
9CB7 20 AA D9                   JSR     IFP
9CBA A2 30                      LDX     #C65536
9CBC A0 B6                      LDY     #C65536/256
9CBE 20 98 DD                   JSR     FLD1R
9CC1 20 8C AB                   JSR     SFMUL
9CC4 20 9C A0                   JSR     FPUSH0
9CC7 A5 BF                      LDA     BINARY+2
9CC9 85 D5                      STA     FR0+1
9CCB A5 C0                      LDA     BINARY+3
9CCD 85 D4                      STA     FR0
9CCF 20 AA D9                   JSR     IFP
9CD2 20 75 A0                   JSR     FPOP1
9CD5 20 AA AB                   JSR     SFADD
9CD8 A5 A9                      LDA     NEGFLG
9CDA 10 06                      BPL     BIN20
9CDC A5 D4                      LDA     FR0             ;NEGATIVE #
9CDE 09 80                      ORA     #$80
```

```
9CE0 85 D4                      STA     FR0
9CE2                    BIN20
9CE2 60                         RTS
```

```
9CE3                    CLNUM                                   ;CLEAR TOKBUF SO NUMBER CAN BE LOADED
9CE3 A2 0F                      LDX     #NUMLEN-1
9CE5 A9 20                      LDA     #' '
9CE7                    CLNLP
9CE7 9D 00 05                   STA     TOKBUF,X
9CEA CA                         DEX
9CEB 10 FA                      BPL     CLNLP
9CED 60                         RTS

9CEE                    DSPCOM
9CEE 20 D2 A2                   JSR     PUTDEL
9CF1 A6 AB                      LDX     NUMFLG
9CF3 D0 04                      BNE     MAIN15
9CF5 A9 11                      LDA     #COLCMD-3       ;# OF BLANKS NEEDED TO GET COMMAND IN PROPER COL
9CF7 D0 06                      BNE     MAIN20          ;JMP     PREVIOUS TOKEN WAS NOT A NUMBER
9CF9                    MAIN15                          ;PREVIOUS TOKEN WAS A NUMBER
9CF9 A9 16                      LDA     #ROWCMD
9CFB 85 54                      STA     ROWCRS
9CFD A9 01                      LDA     #1
9CFF                    MAIN20
9CFF 20 23 A0                   JSR     PUTBLK

9D02 A5 CA                      LDA     DSPFLG
9D04 D0 0D                      BNE     MAIN21          ;NO DISPLAY
9D06 A9 14                      LDA     #COLCMD
9D08 85 55                      STA     COLCRS
9D0A A6 82                      LDX     TOKPTR          ;# OF CHARS
9D0C A9 00                      LDA     #TOKBUF
9D0E A0 05                      LDY     #TOKBUF/256
9D10 20 BA A3                   JSR     PTTXTP
9D13                    MAIN21
9D13 60                         RTS

9D14                    DMEMAL                          ;DISPLAY ALL OF MEM (0-9)
9D14 20 9C A0                   JSR     FPUSH0          ;SAVE X
9D17 A9 09                      LDA     #9
9D19 85 AF                      STA     MEMNUM
9D1B 20 D7 A4           DMELP   JSR     MEMLDR          ;FR0<-MEM(MEMNUM)  (USING MEMNUM)
9D1E 20 89 DD                   JSR     FLDOR
9D21 20 A0 9E                   JSR     TOKNUM          ;TOKBUF<-ASCII(FR0)
9D24 20 2E 9D                   JSR     DSPMEM          ;DISPLAY IN MEM AREA OF SCREEN
9D27 C6 AF                      DEC     MEMNUM
9D29 10 F0                      BPL     DMELP           ;CONTINUE
9D2B 4C 7E A0                   JMP     FPOP0           ;DONE - RELOAD X

9D2E                    DSPMEM                          ;DISPLAY MEMNUM ON SCREEN (ASCII ALREADY IN TOKBUF)
9D2E A5 AF                      LDA     MEMNUM
9D30                    DSPM2                           ;DISPLAY MEM REG A ON SCREEN
9D30 20 47 9D                   JSR     DSPM3
9D33 B0 11                      BCS     DSPM4           ;MEM >= 10 SO DON'T DISPLAY
9D35 A5 CA                      LDA     DSPFLG
9D37 48                         PHA
9D38 A9 00                      LDA     #0
9D3A 85 CA                      STA     DSPFLG
9D3C A9 16                      LDA     #COLCMD+2
9D3E 85 55                      STA     COLCRS
9D40 20 97 9E                   JSR     FDSP2           ;MEM < 10 SO DO DISPLAY
9D43 68                         PLA
9D44 85 CA                      STA     DSPFLG
```

```
9D46                    DSPM4
9D46 60                         RTS

9D47                    DSPM3                           ;SET UP TO DISPLAY MEM REG A
9D47 C9 0A                      CMP     #10
9D49 B0 05                      BCS     DM10            ;ONLY DISPLAY 0-9
9D4B 18                         CLC                     ;ROW = MEMNUM+4
9D4C 69 05                      ADC     #ROWREG
9D4E 85 54                      STA     ROWCRS
9D50                    DM10
9D50 60                         RTS

9D51                    DSPSTK
9D51 A5 CA                      LDA     DSPFLG
9D53 D0 FB                      BNE     DM10            ;DON'T DISPLAY
9D55 20 9C A0                   JSR     FPUSH0          ;DISPLAY STACK    SAVE X ON STACK
9D58 A9 05                      LDA     #ROWREG
9D5A 85 54                      STA     ROWCRS
9D5C A5 A4                      LDA     FPPTR
9D5E 38                         SEC
9D5F E9 06                      SBC     #FPREC
9D61                    STKD10
9D61 85 A8                      STA     T0
9D63 AA                         TAX
9D64 A0 06                      LDY     #FPSTK/256
9D66 20 89 DD                   JSR     FLDOR
9D69 A9 03                      LDA     #3              ;DISPLAY IN COLUMN 3
9D6B 20 92 9E                   JSR     FDSP1

9D6E E6 54                      INC     ROWCRS
9D70 A5 54                      LDA     ROWCRS
9D72 C9 0F                      CMP     #ROWSCR-1
9D74 B0 1C                      BCS     STKD45          ;STACK AT LEAST 10 DEEP
9D76 A5 A8                      LDA     T0
9D78 38                         SEC
9D79 E9 06                      SBC     #FPREC
9D7B B0 E4                      BCS     STKD10          ;CONTINUE
9D7D A5 54                      LDA     ROWCRS
9D7F 48                         PHA                     ;SAVE NEW PRVSTK
9D80                    STKD30
9D80 A5 54                      LDA     ROWCRS          ;CLEAR ALL ROWS UP TO PREVIOUS STACK MAX
9D82 C5 B0                      CMP     PRVSTK
9D84 B0 0B                      BCS     STKD40
9D86 A9 03                      LDA     #3
9D88 85 55                      STA     COLCRS
9D8A 20 58 A3                   JSR     BLNK16
9D8D E6 54                      INC     ROWCRS
9D8F D0 EF                      BNE     STKD30          ;JMP
9D91                    STKD40
9D91 68                         PLA
9D92                    STKD45
9D92 85 B0                      STA     PRVSTK
9D94 4C 7E A0                   JMP     FPOP0
```

```
9D97                    DSPRG                           ;DISPLAY PROGRAM ADDRESS, CURRENT TOKEN
9D97 A5 C7                      LDA     PC+1
9D99 C9 10                      CMP     #PC1MAX+1
9D9B 90 05                      BCC     DG10
9D9D                    EPERR
9D9D A9 BC                      LDA     #EPMSG          ;PAST END OF MEM
9D9F 4C E5 9B                   JMP     ERRSUB

9DA2                    DG10
9DA2 E9 0B                      SBC     #PRGMEM/256-1           ;CARRY CLEAR
9DA4 85 D5                      STA     FR0+1
9DA6 A5 C6                      LDA     PC
9DA8 85 D4                      STA     FR0
9DAA 20 AA D9                   JSR     IFP
9DAD 20 E6 D8                   JSR     FASC            ;0 -> 1023 IN (INBUFF) IN ASCII

9DB0 A0 FF                      LDY     #$FF            ;RIGHT JUSTIFY IN TOKBUF, PUT 0'S AT LEFT
9DB2                    DGLP1
9DB2 C8                         INY                     ;FIND END OF BUFFER
9DB3 B1 F3                      LDA     (INBUFF),Y
9DB5 10 FB                      BPL     DGLP1
9DB7 29 7F                      AND     #$7F            ;MASK OFF END OF BUFFER INDICATOR
9DB9 A2 03                      LDX     #3
9DBB D0 02                      BNE     DG20            ;JMP
9DBD                    DGLP2                           ;MOVE TO TOKBUF
9DBD B1 F3                      LDA     (INBUFF),Y
9DBF                    DG20
9DBF 9D 00 05                   STA     TOKBUF,X
9DC2 CA                         DEX
9DC3 88                         DEY
9DC4 10 F7                      BPL     DGLP2
9DC6 8A                         TXA
9DC7 30 08                      BMI     DG30
9DC9 A9 30                      LDA     #'0             ;PAD WITH 0'S
9DCB                    DGLP3
9DCB 9D 00 05                   STA     TOKBUF,X
9DCE CA                         DEX
9DCF 10 FA                      BPL     DGLP3
9DD1                    DG30
9DD1 20 D2 A2                   JSR     PUTDEL
9DD4 A9 02                      LDA     #LMARG+1
9DD6 85 55                      STA     COLCRS          ;ONE SPACE AT BEGINNING OF LINE
9DD8 A9 00                      LDA     #TOKBUF
9DDA A0 05                      LDY     #TOKBUF/256
9DDC A2 04                      LDX     #4
9DDE 20 93 A3                   JSR     PTCHSP
9DE1 20 3C A2                   JSR     NCHKLD          ;LOAD TOKEN CODE FROM PRGMEM, CHECK FOR NUMBER
9DE4 B0 2E                      BCS     DGRTN           ;ERROR
9DE6 D0 07                      BNE     DG60            ;NOT NUMBER
9DE8 20 A0 9E                   JSR     TOKNUM          ;FP NUMBER -> ASCII IN TOKBUF
9DEB A2 10                      LDX     #NUMLEN
9DED D0 1A                      BNE     DG80            ;JMP

9DEF                    DG60
9DEF 85 81                      STA     TOKCOD          ;NOT A #
9DF1 20 61 A4                   JSR     UNPKEY          ;UNPACK CHARS FOR TOKEN
9DF4 B0 1E                      BCS     DGRTN           ;ERROR

9DF6                    PUTCMD                          ;ENTRY POINT TO DISPLAY COMMAND
9DF6 84 A8                      STY     TO
```

```
9DF8 A9 10              LDA     #NUMLEN
9DFA 38                 SEC
9DFB E5 A8              SBC     TO              ;Y = LENGTH OF BUFFER
9DFD F0 08              BEQ     DG70
9DFF AA                 TAX                     ;OUTPUT BLANKS
9E00 A9 00              LDA     #BLKBUF
9E02 A0 0B              LDY     #BLKBUF/256
9E04 20 93 A3           JSR     PTCHSP
9E07                DG70
9E07 A6 82              LDX     TOKPTR
9E09 A9 00          DG80    LDA     #TOKBUF
9E0B A0 05              LDY     #TOKBUF/256
9E0D 20 93 A3           JSR     PTCHSP
9E10 20 DE A2           JSR     PUTCR           ;CR ON SCREEN ONLY, NOT PRINTER
9E13 18                 CLC
9E14                DGRTN
9E14 60                 RTS                     ;NO ERROR
```

```
9E15                    FPBIN                   ;CONVERT FRO TO 32 BIT BINARY #
                        ;                       THEN COMPARE WITH BITBIN TO SEE IF IT IS IN THE
                        ;                       RANGE SPECIFIED BY BITINT.
9E15 A5 D4                      LDA     FRO
9E17 85 A9                      STA     NEGFLG
9E19 10 04                      BPL     FP10
9E1B 29 7F                      AND     #$7F            ;TAKE ABSOLUTE VALUE
9E1D 85 D4                      STA     FRO
9E1F                    FP10
9E1F 20 2F 9E                   JSR     FPBNCK          ;CONVERT FP TO BINARY
9E22 90 01                      BCC     FP15            ;OVERFLOW
9E24 60                         RTS
9E25                    FP15

9E25 A5 A9                      LDA     NEGFLG
9E27 10 03                      BPL     FP20
9E29 20 1E A5                   JSR     S2CMP           ;NEG # - TAKE COMP, ADD 1
9E2C                    FP20
9E2C 4C 7A 9C                   JMP     BINCHK          ;CHECK TO SEE IF WITHIN RANGE

9E2F                    FPBNCK
9E2F 20 9C A0                   JSR     FPUSH0          ;SAVE COPY OF # FOR MOD
9E32 A2 30                      LDX     #C65536
9E34 A0 B6                      LDY     #C65536/256
9E36 20 89 DD                   JSR     FLDOR
9E39 20 95 A8                   JSR     SMOD            ;FRO = X MOD 65536 (2 LOWER BYTES)
9E3C 20 D2 D9                   JSR     FPI             ;MODFAC = INT(X/65536) (2 UPPER BYTES)
9E3F A5 D4                      LDA     FRO             ;LSB
9E41 85 C0                      STA     BINARY+3
9E43 A5 D5                      LDA     FRO+1
9E45 85 BF                      STA     BINARY+2
9E47 20 34 A0                   JSR     FLDOM           ;LOAD MODFAC
9E4A 20 D2 D9                   JSR     FPI
9E4D B0 09                      BCS     FPERR           ;ERROR: OVERFLOW
9E4F A5 D4                      LDA     FRO
9E51 85 BE                      STA     BINARY+1
9E53 A5 D5                      LDA     FRO+1           ;MSB
9E55 85 BD                      STA     BINARY+0
9E57 60                         RTS

9E58                    FPERR
9E58 A9 36                      LDA     #BOMSG
9E5A 4C E5 9B                   JMP     ERRSUB
```

```
9E5D                    FDSCOM
9E5D 20 73 9E                   JSR     FDSPO           ;DISPLAY FRO FOLLOWED BY "***"
                        ;DAYCOM                         ;ENTRY PT
9E60 C6 AB                      DEC     NUMFLG          ;<- 0
9E62 A9 13                      LDA     #COLCMD-1
9E64 85 55                      STA     COLCRS
9E66 A9 16                      LDA     #ROWCMD
9E68 85 54                      STA     ROWCRS
9E6A A2 04                      LDX     #4              ;# OF CHARS
9E6C A9 C8                      LDA     #STARMS
9E6E A0 B6                      LDY     #STARMS/256
9E70 4C BA A3                   JMP     PTTXTP


9E73                    FDSPO
                        ;                               CONVERT FRO TO ASCII AND DISPLAY
9E73 A9 17                      LDA     #23
9E75 85 54                      STA     ROWCRS
9E77 20 A0 9E                   JSR     TOKNUM          ;CONVERT FRO TO ASCII IN TOKBUF
                        ;DAYDSP                         ;DISPLAY TOKBUF ON SCREEN AND PRINTER (IF ON)
9E7A 20 81 A3                   JSR     PTCRPN          ;PUT CR ON PRINTER IF PREVIOUS THING WAS NUMBER
9E7D A9 03                      LDA     #3
9E7F 85 55                      STA     COLCRS
9E81 A2 10                      LDX     #NUMLEN
9E83 A9 00                      LDA     #TOKBUF
9E85 A0 05                      LDY     #TOKBUF/256
9E87 20 93 A3                   JSR     PTCHSP
9E8A 20 DE A2                   JSR     PUTCR
9E8D A9 01                      LDA     #1
9E8F 85 AB                      STA     NUMFLG
9E91 60                         RTS

9E92                    FDSP1                           ;DISPLAY NUMBER ON SCREEN ONLY, IN COL A
9E92 85 55                      STA     COLCRS
9E94 20 A0 9E                   JSR     TOKNUM
9E97                    FDSP2                           ;DISPLAY TOKBUF NUMBER ON SCREEN WHEREVER CURSOR IS
9E97 A2 10                      LDX     #NUMLEN
9E99 A9 00                      LDA     #TOKBUF
9E9B A0 05                      LDY     #TOKBUF/256
9E9D 4C 5E A3                   JMP     PUTCHS


9EA0                    TOKNUM                  ;CONVERT FRO TO ASCII IN TOKBUF -> TOKBUF+NUMLEN-1 (RIGHT JUSTIFIED)
9EA0 20 9C A0                   JSR     FPUSHO          ;SAVE FP #
9EA3 A5 87                      LDA     DHOFLG
9EA5 F0 40                      BEQ     FDS05

                        ;                                       NON-DECIMAL NUMBER
9EA7 20 55 A8                   JSR     SINTEG          ;TRUNCATE # FOR DISPLAY ONLY
9EAA 20 15 9E                   JSR     FPBIN
9EAD A2 03                      LDX     #3
9EAF                    FDLP3
9EAF B5 BD                      LDA     BINARY,X
9EB1 35 B5                      AND     BITBN2,X
9EB3 95 BD                      STA     BINARY,X
9EB5 CA                         DEX
9EB6 10 F7                      BPL     FDLP3
9EB8 A9 00                      LDA     #0
```

```
9EB6 10 F7              BPL     FDLP3
9EB8 A9 00              LDA     #0
```

COLLEEN CALCULATOR, BY C SHAW

```
9EBA 20 A8 9C           JSR     BINFP2          ;TREAT ALL AS POSITIVE

9EBD 20 E3 9C           JSR     CLNUM           ;CLEAR TOKBUF
9EC0 A9 10              LDA     #NUMLEN
9EC2 85 82              STA     TOKPTR

9EC4                FDLP2
                    ;                           ;PUSH Y ON STACK FOR MOD
9EC4 A5 87              LDA     DHOFLG
9EC6 20 8D A8           JSR     INTMOD
9EC9 20 D2 D9           JSR     FPI
9ECC A5 D4              LDA     FR0             ;0-> (DHOFLG-1)
9ECE 18                 CLC
9ECF 69 30              ADC     #'0
9ED1 C9 3A              CMP     #'0+10
9ED3 90 02              BCC     FDS1
9ED5 69 06              ADC     #'A-10-1-'0
9ED7                FDS1
9ED7 A0 00              LDY     #0
9ED9 C6 82              DEC     TOKPTR
9EDB 91 82              STA     (TOKPTR),Y

9EDD 20 34 A0           JSR     FLDOM           ;LOAD MODFAC=INT(FR0/DHOFLG)
9EE0 A5 D4              LDA     FR0
9EE2 D0 E0              BNE     FDLP2

9EE4 4C 1C A0           JMP     FABCD
```

```
                    ;                           DECIMAL MODE
9EE7                FDS05
9EE7 20 E3 9C           JSR     CLNUM           ;CLEAR TOKBUF
                    ;                           ADD 5X10^K IN PREPARATION FOR ROUNDING
9EEA A2 50              LDX     #$50
9EEC A0 00              LDY     #0
9EEE 84 CD              STY     SEFORM          ;INIT TO INDICATE NOT EFORM
9EF0 A5 D5              LDA     FR0+1
9EF2 85 D1              STA     SMSD            ;SAVE MSD OF NUMBER
9EF4 A5 D4              LDA     FR0
9EF6 85 D0              STA     SSIGN           ;SAVE SIGN
9EF8 F0 65              BEQ     RF90            ;0 => NOTHING FANCY NEEDED
9EFA 29 7F              AND     #$7F            ;TAKE ABSVAL
9EFC 85 D4              STA     FR0
9EFE C9 3F              CMP     #$3F            ;E FORM?
9F00 90 15              BCC     RF10            ;YES.
9F02 C9 44              CMP     #$44
9F04 B0 11              BCS     RF10            ;YES.
9F06 A4 CE              LDY     FIXNUM          ;NO.
9F08 C0 09              CPY     #9              ;FIX 9 (NOFIX)?
9F0A F0 13              BEQ     RF20            ;YES.
9F0C 98                 TYA                     ;FIXNUM
9F0D 4A                 LSR     A               ;DIVIDE BY 2
9F0E 90 02              BCC     RF70            ;ODD?
9F10 A2 05              LDX     #$05            ;YES.
9F12                RF70
9F12 49 3F              EOR     #$3F            ;TAKE COMPLEMENT, CHANGE TO EXCESS 40 NOTATION
9F14 4C 3B 9F           JMP     RF80

9F17                RF10
```

```
9F17 E6 CD              INC     SEFORM          ;YES.  SET FLAG
9F19 A5 CE              LDA     FIXNUM          ;FORM 5X10^-(FIXNUM+1)X10^K1
9F1B C9 08              CMP     #7+1            ;WHERE K1=POWER OF 10 IN FRO.
9F1D 90 02              BCC     RF30            ;MAX # OF DIGITS AFTER DP=7 IN E MODE
9F1F            RF20
9F1F A9 07              LDA     #7
9F21            RF30
9F21 A4 D5              LDY     FRO+1
9F23 C0 10              CPY     #$10            ;IN RANGE 0-9?
9F25 90 04              BCC     RF40            ;YES.
9F27 E9 01              SBC     #1              ;NO. 10-99 => ODD POWER OF 10, 5X10^-((FIXNUM-1)+1)X10^(K1-1)
9F29 C9 FF              CMP     #$FF            ;SET CARRY IF NEG.
9F2B            RF40
9F2B 6A                 ROR     A               ;DIVIDE BY 2 TO GET POWER OF 100
9F2C 90 02              BCC     RF50            ;ODD POWER OF 10?
9F2E A2 05              LDX     #$05            ;YES.  USE 5, NOT $50
9F30            RF50
9F30 49 FF              EOR     #$FF            ;TAKE COMPLEMENT = -(A-1)
9F32 18                 CLC
9F33 65 D4              ADC     FRO             ;COMBINE WITH EXPONENT
9F35 C9 0F              CMP     #$F             ;TOO SMALL?
9F37 90 34              BCC     RFERR           ;YES.  ERROR
9F39 F0 32              BEQ     RFERR           ;YES.  ERROR
9F3B            RF80
9F3B 85 E0              STA     FR1             ;EXPONENT
9F3D 86 E1              STX     FR1+1           ;5 OR $50

9F3F A2 03              LDX     #3
9F41 A9 00              LDA     #0
9F43            RFLP1
9F43 85 E2              STA     FR1+2           ;CLEAR REST OF FR1
9F45 CA                 DEX
9F46 10 FB              BPL     RFLP1

9F48 20 66 DA           JSR     FADD            ;FINALLY CAN DO ADD
9F4B B0 20              BCS     RFERR           ;ERROR - OVERFLOW

9F4D A5 CD              LDA     SEFORM          ;EFORM?
9F4F F0 0E              BEQ     RF90            ;NO.
9F51 20 51 DA           JSR     INTLBF          ;YES.  MAKE INBUFF POINT TO LBUFF
9F54 A9 30              LDA     #'0             ;STORE ASCII 0
9F56 8D 75 05           STA     LBUFF-11
9F59 20 20 D9           JSR     XEFORM          ;FP -> E FORM ASCII
9F5C 4C 62 9F           JMP     RF100
9F5F            RF90
9F5F 20 E6 D8           JSR     FASC            ;FP -> ASCII (NOT EFORM WE HOPE)
9F62            RF100

                ;                               FIND & SAVE E+/-NN
9F62 A0 FF              LDY     #$FF
9F64            RFLP2
9F64 C8                 INY
9F65 B1 F3              LDA     (INBUFF),Y
9F67 10 1B              BPL     RF110
9F69 A6 CD              LDX     SEFORM          ;END OF BUFFER
9F6B F0 0E              BEQ     RF105
9F6D            RFERR                           ;ERROR:  SHOULD HAVE E
9F6D A9 9B              LDA     #CRYMSG         ;OUTPUT ARITHMETIC OVERFLOW MESSAGE
9F6F 20 E5 9B           JSR     ERRSUB
```

```
9F72 20 7E A0            JSR    FPOPO          ;POP OFF OLD X VALUE
9F75 20 9E A2            JSR    PCLRO          ;CLEAR X, DISPLAY 0
9F78 4C A0 9E            JMP    TOKNUM         ;TRY DISPLAY AGAIN

9F7B              RF105
9F7B 29 7F               AND    #$7F           ;CLEAR END OF BUFFER INDICATOR
9F7D 91 F3               STA    (INBUFF),Y
9F7F 84 CF               STY    MANTLN         ;SAVE MANTISSA LENGTH
9F81 4C A0 9F            JMP    RF120

9F84              RF110
9F84 C9 45               CMP    #'E            ;E FOUND?
9F86 D0 DC               BNE    RFLP2          ;NO. CONTINUE

9F88 A6 CD               LDX    SEFORM         ;YES. EFORM?
9F8A F0 E1               BEQ    RFERR          ;NO. ERROR, SHOULD NOT HAVE E
9F8C 88                  DEY
9F8D 84 CF               STY    MANTLN         ;SAVE MANTISSA LENGTH (ADDR OF LAST CHAR)
9F8F C8                  INY
9F90 A2 0C               LDX    #NUMLEN-4      ;MOVE E TO TOKBUF
9F92 9D 00 05     RFLP3  STA    TOKBUF,X
9F95 E8                  INX
9F96 C8                  INY
9F97 B1 F3               LDA    (INBUFF),Y
9F99 10 F7               BPL    RFLP3
9F9B 29 7F               AND    #$7F
9F9D 9D 00 05            STA    TOKBUF,X
9FA0              RF120
                  ;                             NOW FIND DECIMAL PT
9FA0 A0 FF               LDY    #$FF
9FA2              RFLP4
9FA2 C8                  INY
9FA3 C4 CF               CPY    MANTLN
9FA5 F0 0A               BEQ    RF130
9FA7 90 08               BCC    RF130          ;NOT AT END YET
9FA9 A9 2E               LDA    #'.            ;ADD '.'
9FAB 91 F3               STA    (INBUFF),Y
9FAD 84 CF               STY    MANTLN
9FAF D0 06               BNE    RF140          ;JMP
9FB1              RF130
9FB1 B1 F3               LDA    (INBUFF),Y
9FB3 C9 2E               CMP    #'.
9FB5 D0 EB               BNE    RFLP4
9FB7              RF140                        ;HAVE '.'
9FB7 A6 CE               LDX    FIXNUM
9FB9 E0 09               CPX    #9
9FBB D0 22               BNE    RF148          ;FIXED DEC

9FBD A4 CF               LDY    MANTLN         ;NOT FIXED DEC
9FBF C0 09               CPY    #9
9FC1 90 3F               BCC    RF170          ;<= 8 DIGITS IS OK
9FC3 A0 08               LDY    #8
9FC5 A5 D0               LDA    SSIGN          ;LOAD OLD FRO VALUE
9FC7 29 7F               AND    #$7F
9FC9 C9 3F               CMP    #$3F           ;NON EFORM FRACTION (E.G. .1, .01)?
9FCB D0 09               BNE    RF142          ;NO.
9FCD A0 09               LDY    #9             ;YES. ALLOW EXTRA DIGIT FOR LEADING '0'
9FCF A5 D1               LDA    SMSD           ;LOAD OLD FRO+1
9FD1 C9 10               CMP    #$10           ;1 OR 2 DIGITS?
9FD3 B0 01               BCS    RF142          ;2
```

```
9FD5 C8                     INY                     ;1 => ALLOW EXTRA CHAR FOR 0 AFTER D.P.
9FD6                RF142
9FD6 B1 F3                  LDA     (INBUFF),Y
9FD8 C9 30                  CMP     #'0
9FDA D0 26                  BNE     RF170
9FDC 88                     DEY
9FDD 10 F7                  BPL     RF142           ;JMP
9FDF                RF148
9FDF 98                     TYA
9FE0 18                     CLC                     ;COMPUTE WHERE END OF NUMBER SHOULD BE
9FE1 65 CE                  ADC     FIXNUM
9FE3 C5 CF                  CMP     MANTLN
9FE5 B0 04                  BCS     RF150
9FE7 85 CF                  STA     MANTLN          ;MANTISSA TOO LONG => DISCARD DIGITS
9FE9 90 0F                  BCC     RF160           ;JMP
9FEB F0 0D          RF150   BEQ     RF160           ;JUST RIGHT
9FED A4 CF                  LDY     MANTLN          ;MANTISSA TOO SHORT: PAD WITH 0'S
9FEF 85 CF                  STA     MANTLN          ;NEW MANTISSA LENGTH
9FF1 A9 30                  LDA     #'0
9FF3                RFLP5
9FF3 C8                     INY
9FF4 91 F3                  STA     (INBUFF),Y
9FF6 C4 CF                  CPY     MANTLN          ;REACHED DESIRED LENGTH?
9FF8 D0 F9                  BNE     RFLP5           ;NO.  CONTINUE

                    ;                               LIMIT TO 8 DIGITS MAX + DP
9FFA                RF160
9FFA A4 CF                  LDY     MANTLN
9FFC C0 09                  CPY     #9
9FFE 90 02                  BCC     RF170           ;OK
A000 A0 08                  LDY     #8
A002                RF170                           ;MOVE TO TOKBUF
A002 A2 0F                  LDX     #NUMLEN-1
A004 A5 CD                  LDA     SEFORM          ;E FORM?
A006 F0 02                  BEQ     FDLP4           ;NO.
A008 A2 0B                  LDX     #NUMLEN-5       ;YES.  ALLOW ROOM FOR EXPONENT

A00A                FDLP4
A00A B1 F3                  LDA     (INBUFF),Y
A00C 9D 00 05               STA     TOKBUF,X
A00F CA                     DEX
A010 88                     DEY
A011 10 F7                  BPL     FDLP4

                    ;                               CHECK SIGN
A013 A5 D0                  LDA     SSIGN
A015 10 05                  BPL     FABCD
A017 A9 2D                  LDA     #'-             ;NEGATIVE => STORE '-'
A019 9D 00 05               STA     TOKBUF,X
A01C                FABCD
A01C A9 10                  LDA     #NUMLEN
A01E 85 82                  STA     TOKPTR
A020 4C 7E A0               JMP     FPOPO           ;POP ORIGINAL # OFF STACK

A023                PUTBLK
A023 A6 9D                  LDX     PRNFLG          ;PUT BLANKS ON PRINTER ONLY
A025 F0 0C                  BEQ     PUTBRT
A027 A2 20                  LDX     #PIOCB
A029 9D 48 03               STA     ICBLL,X
A02C A9 00                  LDA     #BLKBUF
```

```
A02E A0 0B                      LDY     #BLKBUF/256
A030 4C 66 A3                   JMP     PTCHS2
A033                    PUTBRT
A033 60                         RTS
```

```
A034                    FLDOM                           ;FR0 <- MODFAC
A034 A2 4C                      LDX     #MODFAC
A036 A0 0B                      LDY     #MODFAC/256
A038 4C 89 DD                   JMP     FLDOR
A03B                    FLDOS                           ;FR0 <- TOP OF STACK
A03B 20 8A A0                   JSR     FPOPLD          ;LOAD X&Y REGS WITH STACK POINTER
A03E 4C 89 DD                   JMP     FLDOR
A041                    FLD1S
A041 20 8A A0                   JSR     FPOPLD          ;FR1 <- TOP OF STACK
A044 4C 98 DD                   JMP     FLD1R
A047                    FLDOT                           ;FR0 <- FTEMP
A047 A2 52                      LDX     #FTEMP
A049 A0 0B                      LDY     #FTEMP/256
A04B 4C 89 DD                   JMP     FLDOR

A04E                    FLD1T                           ;FR1 <- FTEMP
A04E A2 52                      LDX     #FTEMP
A050 A0 0B                      LDY     #FTEMP/256
A052 4C 98 DD                   JMP     FLD1R

A055                    FSTOT                           ;FTEMP <- FR0
A055 A2 52                      LDX     #FTEMP
A057 A0 0B                      LDY     #FTEMP/256
A059 4C A7 DD                   JMP     FSTOR
A05C                    FST1T                           ;FTEMP <- FR1
A05C A2 52                      LDX     #FTEMP
A05E A0 0B                      LDY     #FTEMP/256
A060                    FST1R                           ;(X,Y) <- FR1
A060 86 FC                      STX     FLPTR
A062 84 FD                      STY     FLPTR+1
A064 A0 05                      LDY     #5
A066                    FSLOP
A066 B9 E0 00                   LDA     FR1,Y
A069 91 FC                      STA     (FLPTR),Y
A06B 88                         DEY
A06C 10 F8                      BPL     FSLOP
A06E 60                         RTS
A06F                    FMOVE2                          ;FR0 <- FR1
A06F A2 D4                      LDX     #FR0
A071 A0 00                      LDY     #FR0/256
A073 F0 EB                      BEQ     FST1R           ;STORE FR1 IN FR0
A075                    FPOP1
A075 20 84 A0                   JSR     FPOP
A078 4C 98 DD                   JMP     FLD1R           ;FR1 <- POP()
A07B                    FMVPOP
A07B 20 B6 DD                   JSR     FMOVE           ;DO FMOVE THEN FPOP0
A07E                    FPOP0                           FR0 <- POP(FPSTK)
                        ;
A07E 20 84 A0                   JSR     FPOP
A081 4C 89 DD                   JMP     FLDOR
A084                    FPOP
A084 20 8A A0                   JSR     FPOPLD          ;LOAD REGISTERS AND
A087 86 A4                      STX     FPPTR           ;MODIFY STACK POINTER.
A089 60                         RTS
A08A                    FPOPLD                          ;LOAD X & Y REGISTERS IN PREPARATION FOR POP(FPSTK)
A08A A5 A4                      LDA     FPPTR
A08C 38                         SEC
A08D E9 06                      SBC     #FPREC
A08F B0 07                      BCS     FPOP10
```

```
A091 A9 47                  LDA     #NSEMSG
A093 20 E5 9B               JSR     ERRSUB          ;STACK UNDERFLOW
A096 A5 A4                  LDA     FPPTR
A098                FPOP10
A098 AA                     TAX
A099 A0 06                  LDY     #FPSTK/256
A09B 60                     RTS


A09C                FPUSH0                          ;PUSH FR0 ON FPSTK
A09C 20 B3 A0               JSR     FPSHLD
A09F 20 A7 DD               JSR     FSTOR
A0A2                FPSH05
A0A2 A5 A4                  LDA     FPPTR
A0A4 18                     CLC
A0A5 69 06                  ADC     #FPREC
A0A7 85 A4                  STA     FPPTR
A0A9 60                     RTS


A0AA                FPUSH1
A0AA 20 B3 A0               JSR     FPSHLD          ;PUSH FR1 ON FPSTK
A0AD 20 60 A0               JSR     FST1R
A0B0 4C A2 A0               JMP     FPSH05
A0B3                FPSHLD
A0B3 A6 A4                  LDX     FPPTR           ;LOAD REGISTERS & CHECK FOR OVERFLOW
A0B5 E0 FC                  CPX     #FPSLEN*FPREC
A0B7 90 09                  BCC     FPC10
A0B9 A9 53                  LDA     #NSFMSG
A0BB 20 E5 9B               JSR     ERRSUB          ;STACK OVERFLOW
A0BE A2 F6                  LDX     #FPSLEN-1*FPREC
A0C0 86 A4                  STX     FPPTR
A0C2                FPC10
A0C2 A0 06                  LDY     #FPSTK/256
A0C4 60                     RTS
```

```
A0C5                    SUBCAL                                  ;PERFORM OP A (ROUTINE CALLED WILL DO RTS)
A0C5 C9 95                      CMP     #EQUAL+1
A0C7 90 05                      BCC     SBCL5
A0C9 A9 0D                      LDA     #KEYMSG                 ;NOT VALID COMMAND (COULD HAPPEN IN EXECUTE MODE?)
A0CB 4C E5 9B                   JMP     ERRSUB
A0CE                    SBCL5
A0CE 0A                         ASL     A
A0CF A8                         TAY
A0D0 A9 74                      LDA     #JMPTBL
A0D2 85 95                      STA     JMPTR1
A0D4 A9 BB                      LDA     #JMPTBL/256
A0D6 85 96                      STA     JMPTR1+1
A0D8 90 02                      BCC     MAIN10
A0DA E6 96                      INC     JMPTR1+1                ;SECOND PAGE OF TABLE
A0DC                    MAIN10
A0DC B1 95                      LDA     (JMPTR1),Y
A0DE 85 97                      STA     JMPTR2                  ;LOAD AND STORE JSR ADDRESS
A0E0 C8                         INY
A0E1 B1 95                      LDA     (JMPTR1),Y
A0E3 85 98                      STA     JMPTR2+1
A0E5 6C 97 00                   JMP     (JMPTR2)


A0E8                    GTCHR
                        ;                                       RETURN NEXT INPUT CHAR IN A REG
A0E8 A6 86                      LDX     TOKTIN
A0EA F0 07                      BEQ     GETC05

A0EC C6 86                      DEC     TOKTIN
A0EE B5 83                      LDA     TOKTMP-1,X              ;USE CHARS FROM PREVIOUS CALL
A0F0 4C 43 A1                   JMP     GETC10                  ;SAVE CHAR IN TOKBUF

A0F3                    GETC05

A0F3 A2 10                      LDX     #KIOCB
A0F5 A9 07                      LDA     #GETCHR                 ;GET FROM K:    (DATA RETURNED IN A, STATUS IN Y)
A0F7 9D 42 03                   STA     ICCOM,X
A0FA A5 82                      LDA     TOKPTR
A0FC 9D 44 03                   STA     ICBAL,X
A0FF A5 83                      LDA     TOKPTR+1
A101 9D 45 03                   STA     ICBAH,X
A104 A9 01                      LDA     #1
A106 9D 48 03                   STA     ICBLL,X
A109 A9 00                      LDA     #0
A10B 9D 49 03                   STA     ICBLH,X

A10E 20 56 E4                   JSR     CIOV
A111 C0 01                      CPY     #SUCCES
A113 D0 36                      BNE     GETC12                  ;BREAK => DELETE LINE

A115 C9 9B                      CMP     #$9B
A117 90 14                      BCC     GETC06                  ;0-9A
A119 D0 04                      BNE     GNOCR
A11B A9 20                      LDA     #' '                    ;9B
A11D D0 3C                      BNE     GETC30                  ;CR => CHANGE TO BLANK AND DON'T PRINT
A11F                    GNOCR
A11F C9 FD                      CMP     #$FD                    ;FD=BELL
A121 B0 2C                      BCS     GETC15                  ;FD-FF
```

```
A121 B0 2C                      BCS     GETC15          ;FD-FF

COLLEEN CALCULATOR, BY C SHAW


A123 C9 A0                      CMP     #$A0
A125 B0 06                      BCS     GETC06          ;A0-FD
A127 C9 9C                      CMP     #DELLIN         ;9C-9F
A129 F0 2B                      BEQ     GETC20          ;DON'T ESCAPE IF DELETE LINE (9C)
A12B D0 22                      BNE     GETC15          ;9D-9F
A12D                    GETC06                          ;0-9A OR A0-FD
A12D 29 7F                      AND     #$7F            ;STRIP OFF INVERSE VIDEO, IF ANY
A12F F0 1E                      BEQ     GETC15          ;0
A131 C9 1B                      CMP     #$1B
A133 B0 04                      BCS     GETC07
A135 69 40                      ADC     #2*32           ;1-1A (CONVERT CTRL GRAPHICS TO UPPER CASE LETTER)
A137 D0 0A                      BNE     GETC10          ;JMP
A139                    GETC07
A139 C9 61                      CMP     #'A+32          ;LOWER CASE ALPHA TO UPPER CASE
A13B 90 06                      BCC     GETC10
A13D C9 7B                      CMP     #'Z+32+1
A13F B0 02                      BCS     GETC10
A141 E9 1F                      SBC     #32-1           ;CARRY SET

A143                    GETC10
A143 C9 20                      CMP     #'
A145 F0 14                      BEQ     GETC30          ;DON'T PRINT SPACE

A147 C9 7E                      CMP     #BACKSP
A149 D0 04                      BNE     GETC15
A14B                    GETC12
A14B A9 9C                      LDA     #DELLIN
A14D D0 07                      BNE     GETC20          ;JMP    BACKSPACE IS EQUIV TO DELETE LINE
A14F                    GETC15
A14F 48                         PHA
A150 A9 1B                      LDA     #ESC
A152 20 F2 A2                   JSR     PTCHR
A155 68                         PLA
A156                    GETC20
A156 48                         PHA
A157 20 F2 A2                   JSR     PTCHR           ;PUT ON SCREEN
A15A 68                         PLA

A15B                    GETC30
A15B A0 00                      LDY     #0
A15D 91 82                      STA     (TOKPTR),Y
A15F 60                         RTS
```

```
A160                GETDHO
                    ;                                   GET DEC, HEX, OR OCT DIGIT AND RETURN IN A,TOKBUF
A160 20 E8 A0               JSR     GTCHR
A163 C9 9C                  CMP     #DELLIN
A165 D0 05                  BNE     DHOCHK
A167 68                     PLA                         ;IF DELETE LINE THEN POP STACK (SKIP RETURN)
A168 68                     PLA
A169 4C 79 9A               JMP     LEX

A16C                DHOCHK                              ;ENTRY POINT IF ALREADY HAVE CHAR
A16C C9 30                  CMP     #'0
A16E 90 2B                  BCC     DHOERR              ;ERROR
A170 A6 87                  LDX     DHOFLG
A172 E0 08                  CPX     #8
A174 D0 06                  BNE     DH010

A176 C9 38                  CMP     #'7+1               ;OCTAL
A178 B0 21                  BCS     DHOERR
A17A 90 1A                  BCC     DHOOK               ;OK OCT 0-7

A17C                DH010
A17C E0 02                  CPX     #2
A17E D0 06                  BNE     DH020
A180 C9 32                  CMP     #'1+1
A182 B0 17                  BCS     DHOERR
A184 90 10                  BCC     DHOOK
A186                DH020
A186 C9 3A                  CMP     #'9+1
A188 90 0C                  BCC     DHOOK               ;OK DEC OR HEX 0-9
A18A E0 10                  CPX     #16
A18C D0 0D                  BNE     DHOERR              ;ERROR    DEC >9
A18E C9 41                  CMP     #'A
A190 90 09                  BCC     DHOERR
A192 C9 47                  CMP     #'F+1
A194 B0 05                  BCS     DHOERR
A196                DHOOK
A196 E6 82                  INC     TOKPTR              ;SAVE CHAR
A198 A0 00                  LDY     #0                  ;OK         EQ
A19A 60                     RTS
A19B                DHOERR
A19B A0 01                  LDY     #1                  ;NOT OK    NE
A19D 60                     RTS
```

```
A19E                    GETINT                          ;GET INTEGER FROM 0-255 FROM KEYBOARD
                        ;                               USEFUL FOR MEM REG #, FIX, BITS.
                        ;                               RETURN EQ=> OK, NE => NOT OK
A19E A5 87                      LDA     DHOFLG
A1A0 48                         PHA                     ;SAVE DHOFLG
A1A1 A9 00                      LDA     #0              ;FORCE DECIMAL MODE
A1A3 85 87                      STA     DHOFLG
A1A5 20 79 9A                   JSR     LEX
A1A8 A5 81                      LDA     TOKCOD
A1AA C9 96                      CMP     #NUMBER
A1AC F0 06                      BEQ     GI05
A1AE 68                         PLA
A1AF 85 87                      STA     DHOFLG
A1B1 A9 01                      LDA     #1              ;NE
A1B3 60                         RTS
A1B4                    GI05
A1B4 20 D2 A2                   JSR     PUTDEL
A1B7 20 73 9E                   JSR     FDSPO           ;DISPLAY # IN STANDARD LOC
A1BA 68                         PLA
A1BB 85 87                      STA     DHOFLG          ;RESTORE
A1BD                    GINT2                           ;ENTRY PT. IF ALREADY HAVE FRO
A1BD 20 D2 D9                   JSR     FPI
A1C0 B0 04                      BCS     GI10
A1C2 A5 D5                      LDA     FRO+1
A1C4 F0 02                      BEQ     GI20
A1C6                    GI10
A1C6 A9 01                      LDA     #1              ;NE
A1C8                    GI20
A1C8 08                         PHP
A1C9 A5 D4                      LDA     FRO
A1CB 48                         PHA
A1CC 20 47 A0                   JSR     FLDOT
A1CF 68                         PLA                     ;OLD FRO = INTEGER 0-255
A1D0 28                         PLP                     ;EQ OR NE
A1D1 60                         RTS
```

```
A1D2                    GETPRI                  ;INPUT: A=TOKEN CODE.  OUTPUT: A=PRIORITY
A1D2 4A                         LSR     A
A1D3 AA                         TAX
A1D4 BD 29 BB                   LDA     PRIOTB,X
A1D7 B0 04                      BCS     GPR10
A1D9 4A                         LSR     A
A1DA 4A                         LSR     A
A1DB 4A                         LSR     A
A1DC 4A                         LSR     A
A1DD                    GPR10
A1DD 29 0F                      AND     #$F
A1DF A6 99                      LDX     RPNALG
A1E1 E0 02                      CPX     #ALGNOP
A1E3 D0 0A                      BNE     GPR20

A1E5 C9 0D                      CMP     #PHIGH
A1E7 B0 06                      BCS     GPR20
A1E9 C9 06                      CMP     #POR+1
A1EB 90 02                      BCC     GPR20
A1ED A9 05                      LDA     #POR
A1EF                    GPR20
A1EF 60                         RTS

A1F0                    INTDSP                  ;IF INTERM THEN FDSCOM() ELSE DISP=TRUE
A1F0 A5 A0                      LDA     INTERM
A1F2 F0 03                      BEQ     IND10
A1F4 4C 5D 9E                   JMP     FDSCOM
A1F7                    IND10
A1F7 A9 01                      LDA     #1
A1F9 85 A6                      STA     DISP
A1FB 60                         RTS
```

```
A1FC                    LDCHR
                        ;                                       RETURN A=PACKED CHAR
A1FC 20 27 A2                   JSR     LDNIB
A1FF D0 21                      BNE     LDCH10
A201 20 27 A2                   JSR     LDNIB
A204 C6 8A                      DEC     KEYLN2          ;1 EXTRA BYTE
A206 C9 0F                      CMP     #15             ;SPECIAL 4-NIBBLE CHAR?
A208 D0 15                      BNE     LDCH05          ;NO. 2 NIBBLE CHAR
A20A 20 27 A2                   JSR     LDNIB           ;YES. LOAD 2 NIBBLES OF ASCII
A20D 0A                         ASL     A
A20E 0A                         ASL     A
A20F 0A                         ASL     A
A210 0A                         ASL     A
A211 8D 64 0B                   STA     LDCSAV
A214 20 27 A2                   JSR     LDNIB
A217 0D 64 0B                   ORA     LDCSAV          ;COMBINE 2 NIBBLES
A21A C6 8A                      DEC     KEYLN2          ;2 MORE EXTRA BYTES
A21C C6 8A                      DEC     KEYLN2
A21E 60                         RTS                     ;RETURN
A21F                    LDCH05
A21F 18                         CLC                     ;2 NIBBLE CHAR
A220 69 10                      ADC     #16
A222                    LDCH10
A222 AA                         TAX
A223 BD DE B6                   LDA     TABLE-1,X
A226 60                         RTS

A227                    LDNIB
                        ;                                       LOAD PACKED NIBBLE  FROM PKPTR+Y. LFRT
A227 A5 80                      LDA     LFRT
A229 49 01                      EOR     #1
A22B 85 80                      STA     LFRT
A22D F0 08                      BEQ     LDN20
A22F C8                         INY                     ;LEFT NIBBLE
A230 B1 8C                      LDA     (PKPTR),Y
A232 4A                         LSR     A
A233 4A                         LSR     A
A234 4A                         LSR     A
A235 4A                         LSR     A
A236 60                         RTS
A237                    LDN20
A237 B1 8C                      LDA     (PKPTR),Y       ;RIGHT NIBBLE
A239 29 0F                      AND     #$F
A23B 60                         RTS
```

```
A23C                    NCHKLD                                  ;IF TOKEN IS NUMBER THEN LOAD NUMBER INTO FRO FROM PRGMEM
                        ;                                   RETURN EQ => NUMBER, NE => NOT #, CS => ERROR
A23C A0 00                      LDY     #0
A23E B1 C6                      LDA     (PC),Y
A240 85 B1                      STA     TOKCOD
A242 C9 96                      CMP     #NUMBER
A244 D0 1A                      BNE     NCK30
A246 A0 07                      LDY     #FPREC+1
A248 B1 C6                      LDA     (PC),Y
A24A C9 96                      CMP     #NUMBER         ;NUMBER AT OTHER END?
A24C F0 05                      BEQ     NCK10           ;YES
A24E A9 0D                      LDA     #KEYMSG         ;NO.  ERROR
A250 4C E5 9B                   JMP     ERRSUB
A253                    NCK10
A253 20 55 A0                   JSR     FSTOT
A256 20 62 A2                   JSR     PCNCHK          ;SEE IF ROOM LEFT IN PRGMEM FOR #
A259 B0 06                      BCS     NCK40
A25B 20 89 DD                   JSR     FLDOR
A25E A9 00                      LDA     #0              ;EQ
A260                    NCK30
A260 18                         CLC
A261                    NCK40
A261 60                         RTS

A262                    PCNCHK                          ;CHECK PC TO SEE IF ROOM IN PRGMEM FOR #
                        ;                                       RETURN CC => OK, CS=> NOT OK
A262 A6 C6                      LDX     PC
A264 A4 C7                      LDY     PC+1
A266 C0 0F                      CPY     #PC1MAX
A268 90 07                      BCC     PCN10
A26A E0 F9                      CPX     #-FPREC-1
A26C 90 03                      BCC     PCN10
A26E                    PCN05
A26E 4C 9D 9D                   JMP     EPERR
A271                    PCN10
A271 E8                         INX
A272 D0 01                      BNE     PCN20
A274 C8                         INY
A275                    PCN20
A275 60                         RTS


A276                    PCINC                           ;PC <- PC+1
A276 A9 01                      LDA     #1
A278 D0 02                      BNE     PCADD           ;JMP
A27A                    PCADDN
A27A A9 08                      LDA     #FPREC+2        ;PC <- PC+FPREC+2
A27C                    PCADD
A27C 18                         CLC
A27D 65 C6                      ADC     PC
A27F 90 09                      BCC     PCADD1
A281 A6 C7                      LDX     PC+1            ;INC MSB
A283 E8                         INX
A284 E0 10                      CPX     #PC1MAX+1       ;END OF MEM?
A286 B0 E6                      BCS     PCN05           ;YES.  DON'T CHANGE PC
A288 86 C7                      STX     PC+1            ;NO.  STORE NEW PC
A28A                    PCADD1
A28A 85 C6                      STA     PC              ;STORE LSB
A28C 60                         RTS                     ;RETURN CARRY CLEAR => NO ERROR
```

```
A28D                    INTCMP                          ;FTEMP <- FR0   ;FR0 <- FR0 - FP(A)
A28D 48                         PHA
A28E 20 55 A0                   JSR     FSTOT
A291 68                         PLA
A292 20 A7 A2                   JSR     PSET0
A295                    INTC2
A295 20 B6 DD                   JSR     FMOVE
A298 20 47 A0                   JSR     FLDOT
A29B 4C B9 AB                   JMP     SFSUB

A29E                    PCLR0                           ;CLEAR FR0
A29E A9 00                      LDA     #0
A2A0 F0 05                      BEQ     PSET0           ;JMP

A2A2                    LDINT                           ;MOVE FR0 TO FR1, THEN SET FR0 TO A
A2A2 48                         PHA
A2A3 20 B6 DD                   JSR     FMOVE           ;FR1 <- FR0
A2A6 68                         PLA

A2A7                    PSET0                           ;SET FR0 TO INTEGER PASSED IN A
A2A7 85 D4                      STA     FR0
A2A9 A9 00                      LDA     #0
A2AB 85 D5                      STA     FR0+1
A2AD 4C AA D9                   JMP     IFP             ;INTEGER A TO FP A

A2B0                    POPOP                           ;POP A OFF OPSTK
A2B0 A6 A5                      LDX     OPPTR
A2B2 D0 06                      BNE     POP10
A2B4 A9 5E                      LDA     #OSEMSG
A2B6 20 E5 9B                   JSR     ERRSUB          ;STACK UNDERFLOW
A2B9 E8                         INX
A2BA                    POP10
A2BA CA                         DEX
A2BB 86 A5                      STX     OPPTR
A2BD BD 00 07                   LDA     OPSTK,X
A2C0 60                         RTS

A2C1                    PUSHOP                          ;PUSH A ON OPSTK
A2C1 A6 A5                      LDX     OPPTR
A2C3 9D 00 07                   STA     OPSTK,X
A2C6 E8                         INX
A2C7 D0 06                      BNE     PSH10
A2C9 A9 6A                      LDA     #OSFMSG
A2CB 20 E5 9B                   JSR     ERRSUB          ;STACK OVERFLOW
A2CE CA                         DEX
A2CF                    PSH10
A2CF 86 A5                      STX     OPPTR
A2D1 60                         RTS
```

```
A2D2                    PUTDEL                                  ;DELETE BOTTOM LINE ON SCREEN
A2D2 A9 17                      LDA     #23
A2D4 85 54                      STA     ROWCRS
A2D6                    PTDEL2                                  ;DELETE CURRENT LINE
A2D6 A9 01                      LDA     #LMARG
A2D8 85 55                      STA     COLCRS
A2DA A9 9C                      LDA     #DELLIN
A2DC D0 14                      BNE     PTCHR                   ;JMP
A2DE                    PUTCR
A2DE A9 9B                      LDA     #CR
A2E0 A6 54                      LDX     ROWCRS
A2E2 E0 17                      CPX     #23
A2E4 D0 0C                      BNE     PTCHR                   ;IF NOT ON BOTTOM LINE THEN DO NORMAL CR
                        ;                                       ;OTHERWISE, DO SCROLLING & RETURN TO BOTTOM LINE
A2E6 A9 10                      LDA     #ROWSCR
A2E8 85 54                      STA     ROWCRS
A2EA 20 D6 A2                   JSR     PTDEL2
A2ED A9 17                      LDA     #23
A2EF 85 54                      STA     ROWCRS
A2F1                    RETN1
A2F1 60                         RTS

A2F2                    PTCHR
                        ;                                       PUT ONE CHAR (IN A) ON SCREEN
A2F2 A2 00                      LDX     #SIOCB
A2F4                    PTCHR2
A2F4 A8                         TAY
A2F5 A5 CA                      LDA     DSPFLG
A2F7 F0 03                      BEQ     PTABC
A2F9 A0 01                      LDY     #SUCCES                 ;DON'T PRINT => ALWAYS SUCCESSFUL
A2FB 60                         RTS
A2FC                    PTABC
A2FC A9 0B                      LDA     #PUTCHR
A2FE 9D 42 03                   STA     ICCOM,X
A301 A9 00                      LDA     #0
A303 9D 48 03                   STA     ICBLL,X
A306 9D 49 03                   STA     ICBLH,X
A309 98                         TYA

A30A 4C 56 E4                   JMP     CIOV


A30D                    CHSTAT                                  ;CHANGE STATUS BY DISPLAYING TOKBUF AT A,0
A30D 85 55                      STA     COLCRS
A30F A5 CA                      LDA     DSPFLG
A311 48                         PHA
A312 A9 00                      LDA     #0
A314 85 CA                      STA     DSPFLG                  ;ALWAYS DISPLAY STATUS
A316 A2 01                      LDX     #ROWSTT
A318 86 54                      STX     ROWCRS
A31A A6 82                      LDX     TOKPTR
A31C E0 04                      CPX     #4
A31E B0 06                      BCS     CHS30
A320 A9 20                      LDA     #'                      ;ADD ONE BLANK TO CLEAR LONGER WORDS
A322 9D 00 05                   STA     TOKBUF,X
A325 E8                         INX
A326                    CHS30
A326 A9 00                      LDA     #TOKBUF
A328 A0 05                      LDY     #TOKBUF/256
A32A 20 5E A3                   JSR     PUTCHS
```

```
A32D 68                 PLA
A32E 85 CA              STA     DSPFLG
A330 60                 RTS


A331                    PTLIN1                          ;PUT UP ONE LINE OF SCREEN DISPLAY (FOR INIT)
A331 86 A8                      STX     TO
A333 BD CC B6                   LDA     CHRTAB,X
A336 20 F2 A2                   JSR     PTCHR
A339 20 50 A3                   JSR     CTLR17
A33C A6 A8                      LDX     TO
A33E BD CD B6                   LDA     CHRTAB+1,X
A341 20 F2 A2                   JSR     PTCHR
A344 A2 12                      LDX     #18
A346 20 52 A3                   JSR     CTLR
A349 A6 A8                      LDX     TO
A34B BD CE B6                   LDA     CHRTAB+2,X
A34E D0 A2                      BNE     PTCHR           ;JMP

A350                    CTLR17                  ;PUT 17 CTRL R'S ON SCREEN (HORIZ. LINES)
A350 A2 11                      LDX     #17
A352                    CTLR                            ;PUT X CTRL R'S ON SCREEN
A352 A9 26                      LDA     #CTLRS
A354 A0 0B                      LDY     #CTLRS/256
A356 D0 06                      BNE     PUTCHS          ;JMP

A358                    BLNK16                          ;PUT 16 BLANKS ON SCREEN
A358 A2 10                      LDX     #16
A35A                    BLNKS                           ;PUT X BLANKS ON SCREEN
A35A A9 00                      LDA     #BLKBUF
A35C A0 0B                      LDY     #BLKBUF/256

A35E                    PUTCHS
                        ;                               A=ICBAL, Y=ICBAH, X=# OF CHARS
A35E 48                 PHA
A35F 8A                 TXA
A360 A2 00              LDX     #SIOCB
A362 9D 48 03           STA     ICBLL,X
A365 68                 PLA
A366                    PTCHS2
A366 9D 44 03           STA     ICBAL,X
A369 A5 CA              LDA     DSPFLG
A36B F0 03              BEQ     PTABD
A36D A0 01              LDY     #SUCCES         ;DON'T PRINT => ALWAYS SUCCESSFUL
A36F 60                 RTS
A370                    PTABD
A370 98                 TYA
A371 9D 45 03           STA     ICBAH,X
A374 A9 00              LDA     #0
A376 9D 49 03           STA     ICBLH,X
A379 A9 0B              LDA     #PUTCHR
A37B 9D 42 03           STA     ICCOM,X

A37E 4C 56 E4           JMP     CIOV

A381                    PTCRPN                          ;IF PREVIOUS TOKEN WAS NUMBER THEN PUT CR ON PRINTER
A381 A5 AB              LDA     NUMFLG
A383 F0 34              BEQ     RETN2
A385                    PUTCRP                          ;PUT CR ON PRINTER IF PRINTER IS ON
A385 A6 9D              LDX     PRNFLG
```

```
A387 F0 30                  BEQ     RETN2
A389 A9 9B                  LDA     #CR
A38B A2 20                  LDX     #PIOCB
A38D 20 F4 A2               JSR     PTCHR2
A390 4C AE A3               JMP     PRNCHK          ;CHECK TO SEE IF PRINTER STILL THERE

A393                PTCHSP                          ;PUT CHARS ON SCREEN AND PRINTER (IF OPEN)
                    ;                               A=ICBAL, Y=ICBAH, X=# OF CHARS
A393 85 AC                  STA     ASAVE
A395 86 AD                  STX     XSAVE
A397 84 AE                  STY     YSAVE
A399 20 5E A3               JSR     PUTCHS
A39C A6 9D                  LDX     PRNFLG
A39E F0 19                  BEQ     RETN2
A3A0 A2 20                  LDX     #PIOCB
A3A2 A5 AD                  LDA     XSAVE
A3A4 9D 48 03               STA     ICBLL,X
A3A7 A5 AC                  LDA     ASAVE
A3A9 A4 AE                  LDY     YSAVE
A3AB 20 66 A3               JSR     PTCHS2
A3AE                PRNCHK
A3AE C0 01                  CPY     #SUCCES         ;SUCCESSFUL PRINTING?
A3B0 F0 07                  BEQ     RETN2           ;YES.
A3B2 C0 80                  CPY     #$80            ;BREAK KEY ABORT?
A3B4 F0 03                  BEQ     RETN2           ;YES.  OK - WILL BE HANDLED LATER
A3B6 4C 09 A9               JMP     OFFERR          ;NO.  CLOSE PRINTER &DISPLAY ERROR MSG
A3B9                RETN2
A3B9 60                     RTS

A3BA                PTTXTP                          ;PUT TEXT ON SCREEN AND PRINTER (IF OPEN)
A3BA 20 93 A3               JSR     PTCHSP
A3BD 20 85 A3               JSR     PUTCRP
A3C0 20 DE A2               JSR     PUTCR
A3C3 60                     RTS
```

```
A3C4                    RAMCLR
A3C4 A9 00              RAMSET  LDA     #0
A3C6 48                         PHA
A3C7 84 8D                      STY     CLRPTR+1
A3C9 86 A8                      STX     TO              ;MEM UPPER LIMIT
A3CB A9 00                      LDA     #0
A3CD 85 8C                      STA     CLRPTR
A3CF A8                         TAY
A3D0 68                         PLA
A3D1                    INIT3
A3D1 91 8C                      STA     (CLRPTR),Y
A3D3 C8                         INY
A3D4 D0 FB                      BNE     INIT3
A3D6 E6 8D                      INC     CLRPTR+1
A3D8 A6 8D                      LDX     CLRPTR+1
A3DA E4 A8                      CPX     TO
A3DC D0 F3                      BNE     INIT3
A3DE 60                         RTS

A3DF                    SAVCHR
                        ;                       MOVE CHAR FROM TOKBUF TO TOKTMP
A3DF A0 00                      LDY     #0
A3E1 B1 82                      LDA     (TOKPTR),Y
A3E3 A6 86                      LDX     TOKTIN
A3E5 95 84                      STA     TOKTMP,X
A3E7 E6 86                      INC     TOKTIN
A3E9 60                         RTS

A3EA                    SAVMAT
A3EA E4 94                      CPX     KMATCH          ;SAVE LONGEST MATCH
A3EC 90 0D                      BCC     KEY30
A3EE F0 0B                      BEQ     KEY30
A3F0 86 94                      STX     KMATCH
A3F2 A2 03                      LDX     #3
A3F4                    KEY25
A3F4 B5 8C                      LDA     PKPTR,X
A3F6 95 90                      STA     KYPTSV,X
A3F8 CA                         DEX
A3F9 10 F9                      BPL     KEY25

A3FB                    KEY30
A3FB 60                         RTS
```

```
A3FC                    UNPINT
A3FC A9 63                      LDA     #KEYWRD-1               ;ALPHA    GET REST OF WORD
A3FE 85 8C                      STA     PKPTR
A400 A9 B9                      LDA     #KEYWRD/256
A402 85 8D                      STA     PKPTR+1
A404 A9 00                      LDA     #0
A406 85 94                      STA     KMATCH
A408 85 80                      STA     LFRT
A40A 85 8E                      STA     KYLFRT
A40C 85 8F                      STA     KEYCNT
A40E 60                         RTS
A40F                    UNPNUM
A40F A0 00                      LDY     #0                      ;SET UP Y REG FOR LDNIB
A411 84 88                      STY     KEYCHR
A413 20 27 A2                   JSR     LDNIB

A416 85 89                      STA     KEYLEN
A418 85 8A                      STA     KEYLN2
A41A 60                         RTS
A41B                    UNPNXT
A41B E6 8F                      INC     KEYCNT
A41D E6 89                      INC     KEYLEN
A41F A5 89                      LDA     KEYLEN
A421 4A                         LSR     A
A422 AA                         TAX
A423 A5 8E                      LDA     KYLFRT
A425 90 07                      BCC     KEY50
A427 49 01                      EOR     #1
A429 F0 01                      BEQ     KEY40
A42B E8                         INX
A42C                    KEY40
A42C 85 8E                      STA     KYLFRT
A42E                    KEY50
A42E 85 80                      STA     LFRT
A430 8A                         TXA
A431 18                         CLC
A432 65 8C                      ADC     PKPTR
A434 90 02                      BCC     KEY60
A436 E6 8D                      INC     PKPTR+1
A438                    KEY60
A438 85 8C                      STA     PKPTR
A43A 60                         RTS
A43B                    UNPCK2
A43B 48                         PHA
A43C 4C 4A A4                   JMP     UNP10
A43F                    UNPACK

                        ;                               UNPACK WORD INTO TOKBUF,X FROM (PKPTR),LFRT
                        ;                               RETURNS LENGTH OF WORD IN Y

A43F 86 82                      STX     TOKPTR
A441 A0 00                      LDY     #0
A443 98                         TYA
A444 48                         PHA
A445 20 27 A2                   JSR     LDNIB
A448 85 8A                      STA     KEYLN2
A44A                    UNP10
A44A 20 FC A1                   JSR     LDCHR
A44D 84 8B                      STY     LDNBSV
A44F AA                         TAX
A450 68                         PLA
A451 A8                         TAY
```

```
A452 8A                 TXA
A453 91 82              STA     (TOKPTR),Y
A455 C8                 INY
A456 98                 TYA
A457 48                 PHA
A458 A4 8B              LDY     LDNBSV
A45A C6 8A              DEC     KEYLN2
A45C D0 EC              BNE     UNP10
A45E 68                 PLA
A45F A8                 TAY
A460 60                 RTS


A461            UNPKEY                          ;UNPACK KEYWORD GIVEN TOKEN CODE IN TOKCOD
                ;                                       ;OUTPUT: CHARS IN TOKBUF, Y=TOKPTR, CS IF ERROR
A461 A5 81              LDA     TOKCOD
A463 C9 8E              CMP     #STAR
A465 90 0F              BCC     UNKY10
A467 C9 95              CMP     #EQUAL+1                ;OUT OF RANGE?
A469 B0 2D              BCS     UKERR           ;YES.
A46B AA                 TAX                     ;NO.  SPECIAL CHAR
A46C BD 0E B6           LDA     TOKCHR-STAR,X
A46F 8D 00 05           STA     TOKBUF
A472 A0 01              LDY     #1
A474 D0 0E              BNE     UNKRTN          ;JMP
A476            UNKY10
A476 20 FC A3           JSR     UNPINT          ;INITIALIZE
A479            UNPLP
A479 A5 8F              LDA     KEYCNT
A47B C5 81              CMP     TOKCOD
A47D D0 0E              BNE     UNKY20
A47F A2 00              LDX     #TOKBUF
A481 20 3F A4           JSR     UNPACK
A484            UNKRTN
A484 84 82              STY     TOKPTR
A486 A9 9B              LDA     #CR
A488 99 00 05           STA     TOKBUF,Y
A48B 18                 CLC                     ;NO ERROR
A48C 60                 RTS

A48D            UNKY20                          ;CONTINUE WITH NEXT WORD
A48D 20 0F A4           JSR     UNPNUM
A490 F0 06              BEQ     UKERR           ;END OF LIST => ERROR (SHOULDN'T HAPPEN)
A492 20 1B A4           JSR     UNPNXT
A495 4C 79 A4           JMP     UNPLP

A498            UKERR
A498 A9 0D              LDA     #KEYMSG
A49A 4C E5 9B           JMP     ERRSUB
```

```
A49D                    ARCSUB
A49D 20 9C A0                   JSR     FPUSHO                  ;FR1 <- SQRT(1-FRO*FRO)  FOR ARCCOS, ARCSIN
A4A0 20 AB AA                   JSR     SSQUAR
A4A3 A9 01                      LDA     #1
A4A5 20 B0 AB                   JSR     INTSUB
A4A8 20 BA B4                   JSR     SSQRT
A4AB 20 B6 DD                   JSR     FMOVE
A4AE 4C 7E A0                   JMP     FPOPO


A4B1                    GETMN
A4B1 A9 45                      LDA     #MEMMSG                 ;FETCH & STORE MEMNUM
A4B3 20 6A 9C                   JSR     PUTMSG                  ;DISPLAY "ENTER MEMORY REGISTER 0-99"
A4B6 20 9E A1                   JSR     GETINT
A4B9 D0 09                      BNE     GMERR                   ;ERROR
A4BB C9 64                      CMP     #MEMLEN
A4BD B0 05                      BCS     GMERR                   ;ERROR
A4BF 85 AF                      STA     MEMNUM                  ;OK  0 -> MEMLEN-1
A4C1 A2 00                      LDX     #0                      ;EQ => OK
A4C3 60                         RTS
A4C4                    GMERR
A4C4 A9 76                      LDA     #BITMSG
A4C6 4C E5 9B                   JMP     ERRSUB                  ;DISPLAY ERROR MESSAGE (WILL RETURN WITH NE => ERROR)


A4C9                    MEMLD0                                  ;FRO <- MEM(A)
A4C9 20 D5 A4                   JSR     MEMLD2
A4CC 4C 89 DD                   JMP     FLDOR
A4CF                    MEMLD1                                  ;FR1 <- MEM(A)
A4CF 20 D5 A4                   JSR     MEMLD2
A4D2 4C 98 DD                   JMP     FLD1R
A4D5                    MEMLD2
A4D5 85 AF                      STA     MEMNUM                  ;SET UP X & Y REGS TO LOAD OR STORE MEM(A)
A4D7                    MEMLDR                                  ;SET UP X & Y REGS TO LOAD OR STORE MEM(MEMNUM)
A4D7 A0 08                      LDY     #MEMORY/256
A4D9 A5 AF                      LDA     MEMNUM                  ;MEMNUM <- MEMNUM*6     (FPREC=6)
A4DB 0A                         ASL     A
A4DC 65 AF                      ADC     MEMNUM                  ;(CARRY IS CLEAR)
A4DE 90 01                      BCC     MLD10
A4E0 C8                         INY
A4E1                    MLD10
A4E1 0A                         ASL     A
A4E2 90 01                      BCC     MLD20
A4E4 C8                         INY
A4E5                    MLD20
A4E5 AA                         TAX
A4E6 60                         RTS


A4E7                    MEMSUB                                  ;SET UP FOR DIV, PRD, SUB, SUM, XCHM, SRCL
A4E7 20 B1 A4                   JSR     GETMN                   ;GET MEMNUM
A4EA F0 04                      BEQ     MS10
A4EC 68                         PLA                             ;ERROR => RETURN 2 LEVELS UP
A4ED 68                         PLA
A4EE 38                         SEC             ;INDICATE ERROR
A4EF 60                         RTS
A4F0                    MS10
A4F0 20 9C A0                   JSR     FPUSHO                  ;SAVE X ON STACK
A4F3 20 B6 DD                   JSR     FMOVE                   ;FR1 <- X
A4F6 20 D7 A4                   JSR     MEMLDR                  ;SET UP X & Y REGS
```

```
A4F6 20 D7 A4                   JSR     MEMLDR          ;SET UP X & Y REGS
```

COLLEEN CALCULATOR, BY C SHAW

```
A4F9 20 89 DD                   JSR     FLDOR           ;FRO <- MEM(MEMNUM)
A4FC 18                         CLC                     ;INDICATE NO ERROR
A4FD 60                         RTS
A4FE                    MEMMUL                          ;FRO <- FRO*MEM(A)
A4FE 85 AF                      STA     MEMNUM
A500 20 D7 A4                   JSR     MEMLDR
A503 20 98 DD                   JSR     FLD1R
A506 4C 8C AB                   JMP     SFMUL
```

```
                .IF     ASMBL           ;THIS CODE LEFT OUT IF ASMBL=0
;                                       INPUT: FRO IN MMDD.YYYY FORMAT
;                                       Z = YYYY; IF (MM-1) <=1 THEN Z=Z-1;
;                                       OUTPUT:FRO = FACTOR = 365*YYYY + DD + 31*(MM-1) - (DAYTRM,(MM-1))
;                                                            +INT(Z/4) - INT(.75[INT(Z/100)+1])
DAYERR
        LDA     #DAYMSG
        JSR     ERRSUB
        JSR     PCLR0           ;CLEAR X
        SEC                     ;INDICATE ERROR
        RTS

DAYSUB
        LDA     FR0
        BMI     DAYERR          ;MUST BE >0
        JSR     FSTOT
        JSR     SINTEG
;                               ;COMPUTE DD = X MOD 100
        LDA     #100
        JSR     INTMOD

        LDA     MODFAC+1        ;CHECK MM AND DD
        BEQ     DAYERR          ;MM = 0 => ERROR
        CMP     #$13
        BCS     DAYERR          ;MM > 12 => ERROR
        SED                     ;MM <- MM-1    (0-$11)
        SBC     #1-1
        STA     MODFAC+1
        CLD
        CMP     #$10            ;DEC -> INTEGER
        BCC     DAYS10
        SBC     #6
DAYS10
        TAX
        LDA     FR0+1           ;DD = 0 => ERROR
        BEQ     DAYERR
        CMP     MAXDAY,X
        BCS     DAYERR          ;DD TOO LARGE
        LDA     DAYTRM,X
        STA     DAYTMP          ;SAVE INT(.4MM+2.3)

        JSR     FPUSH0          ;PUSH DD
        JSR     FLDOT           ;YYYY <- FRACT(X) * 10000
        JSR     SFRACT
        LDX     #C10000
        LDY     #C10000/256
        JSR     FLD1R
        JSR     SFMUL
        JSR     SINTEG
        LDA     FR0
        CMP     #$41
        BNE     DAYERR          ;MUST HAVE $41,YY,YY,0,0,0,0
        LDA     FR0+1
        CMP     #$16
        BCC     DAYERR          ;YYYY MUST BE GE 1600
        JSR     FSTOT           ;FTEMP <- YYYY

        LDX     #C365           ;365 * YYYY
        LDY     #C365/256
```

```
                LDY     #C3657256

                JSR     FLD1R
                JSR     SFMUL

                JSR     FPOP1           ;DD
                JSR     SFADD           ;365*YYYY + DD

                LDX     #MODFAC
                LDY     #MODFAC/256     ;LOAD MM-1
                JSR     FLD1R
                JSR     FPUSH0
                LDA     #31
                JSR     PSET0
                JSR     SFMUL           ;31*(MM-1)
                JSR     FPOP1
                JSR     SFADD           ;365*YYYY + DD + 31*(MM-1)

                LDA     MODFAC+1
                CMP     #2              ;JAN OR FEB?
                BCS     DAYS20          ;NO.
                LDA     FTEMP+2         ;YES.  YYYY <- YYYY-1
                SED
                SBC     #1-1            ;CARRY IS CLEAR
                STA     FTEMP+2
                BCS     DAYS15
                LDA     FTEMP+1
                SBC     #1-1
                STA     FTEMP+1
DAYS15
                CLD
DAYS20
                JSR     FMOVE           ;ADD -(DAYTRM,(MM-1))
                LDA     DAYTMP
                JSR     PSET0
                JSR     SCHGSG
                JSR     SFADD

                JSR     FPUSH0          ;ADD INT(YYYY/4)
                LDA     #4
                JSR     PSET0
                JSR     FMOVE
                JSR     FLDOT
                JSR     SFDIV
                JSR     SINTEG
                JSR     FPOP1
                JSR     SFADD

                JSR     FPUSH0          ;SUB INT(.75[INT[(YYYY/100)+1])
                LDA     #100
                JSR     PSET0
                JSR     FMOVE
                JSR     FLDOT           ;YYYY
                JSR     SFDIV           ;YYYY/100
                LDA     #1
                JSR     INTADD          ;1+YYYY/100
                JSR     SINTEG
                LDX     #CPT75
                LDY     #CPT75/256
                JSR     FLD1R
                JSR     SFMUL
                JSR     SINTEG
```

```
                        JSR     FMOVE
                        JSR     FPOPO
                        JMP     SFSUB


HYPSUB                                  ;FRO <- EXPE(X), FR1<- EXPE(-X)
;                                               FOR COSH, SINH
                        JSR     FPUSHO
                        JSR     SCHGSG
                        JSR     SEXPE           ;EXPE(-X)
                        JSR     SXCHGY
                        JSR     SEXPE           ;EXPE(X)
                        JMP     FPOP1
                        .ENDIF
```

```
A509                    PIOVL           ;LOAD X & Y REGS IN PREPARATION FOR LOADING REG 0 OR 1 WITH PI/2, 90 OR 100(IF GRAD)
A509 A9 24                      LDA     #RADPI2
A50B 18                         CLC
A50C 65 FB                      ADC     RADFLG
A50E AA                         TAX
A50F A0 B6                      LDY     #RADPI2/256
A511 60                         RTS

A512                    SCMP2                           ;TAKE COMPLEMENT OF BINARY
A512 A2 03                      LDX     #3
A514                    SCLP2
A514 B5 BD                      LDA     BINARY,X
A516 49 FF                      EOR     #$FF
A518 95 BD                      STA     BINARY,X
A51A CA                         DEX
A51B 10 F7                      BPL     SCLP2
A51D 60                         RTS


A51E                    S2CMP                           ;TAKE COMPLEMENT OF BINARY AND ADD 1
A51E 20 12 A5                   JSR     SCMP2
A521 E6 C0                      INC     BINARY+3
A523 D0 0A                      BNE     STCRTN
A525 E6 BF                      INC     BINARY+2
A527 D0 06                      BNE     STCRTN
A529 E6 BE                      INC     BINARY+1
A52B D0 02                      BNE     STCRTN
A52D E6 BD                      INC     BINARY+0
A52F                    STCRTN
A52F 60                         RTS


A530                    SLSHF2                          ;SHIFT BINARY LEFT A PLACES
A530 08                         PHP                     ;ROTATING IN CARRY
A531 AA                         TAX
A532 F0 0D                      BEQ     SRTN
A534                    SLS05
A534 28                         PLP
A535 08                         PHP
A536 26 C0                      ROL     BINARY+3
A538 26 BF                      ROL     BINARY+2
A53A 26 BE                      ROL     BINARY+1
A53C 26 BD                      ROL     BINARY+0
A53E CA                         DEX
A53F D0 F3                      BNE     SLS05
A541                    SRTN
A541 28                         PLP
A542 60                         RTS
```

A543 SNUMB

```
                        ;                                       NUMBER PROCESSING:  CONVERT ASCII IN TOKBUF TO FP IN FRO
A543 A5 87                      LDA     DHOFLG
A545 F0 62                      BEQ     SNUM40          ;DECIMAL

A547 A9 FF                      LDA     #-1
A549 85 82                      STA     TOKPTR
A54B 20 9E A2                   JSR     PCLRO           ;HEX BINARY OR OCT => CONVERT TO F.P.
A54E                SNUM20
A54E E6 82                      INC     TOKPTR
A550 A0 00                      LDY     #0
A552 B1 82                      LDA     (TOKPTR),Y
A554 C9 9B                      CMP     #CR
A556 D0 3B                      BNE     SNUM25          ;CONTINUE

                        ;               IS # > BITBIN (2^(BITINT-1)-1) AND <= BITBN2 (2^BITINT)-1?
A558 20 9C A0                   JSR     FPUSHO          ;SAVE FP #
A55B 20 2F 9E                   JSR     FPBNCK          ;CONVERT FP TO BINARY (4 BYTES)
A55E 20 7E A0                   JSR     FPOPO           ;RESTORE FP #
A561 A2 00                      LDX     #0
A563                BBLP1
A563 B5 BD                      LDA     BINARY,X
A565 D5 B1                      CMP     BITBIN,X
A567 F0 04                      BEQ     BB05            ;OK SO FAR
A569 B0 09                      BCS     BB10            ;> BITBIN => OK
A56B 90 4B                      BCC     SNUM50          ;< BITBIN => RETURN
A56D                BB05
A56D E8                         INX
A56E E0 04                      CPX     #4
A570 D0 F1                      BNE     BBLP1           ;CONTINUE
A572 F0 44                      BEQ     SNUM50          ;= BITBIN => RETURN
A574                BB10
A574 A2 00                      LDX     #0
A576                BBLP2
A576 B5 BD                      LDA     BINARY,X
A578 D5 B5                      CMP     BITBN2,X
A57A F0 04                      BEQ     BB20            ;OK SO FAR
A57C B0 3A                      BCS     SNUM50          ;> BITBN2 => RETURN
A57E 90 05                      BCC     BB30            ;< BITBN2 => OK
A580                BB20
A580 E8                         INX
A581 E0 04                      CPX     #4
A583 D0 F1                      BNE     BBLP2
                        ;                               ;= BITBN2 => OK
A585                BB30
A585 A2 03                      LDX     #3              ;OK => INPUT WAS REALLY MEANT AS NEG. #
A587                BBLP3
A587 B5 BD                      LDA     BINARY,X        ;OR WITH BINMIN= -(2^(BITNIT-1)) TO EXTEND SIGN BIT
A589 15 B9                      ORA     BINMIN,X
A58B 95 BD                      STA     BINARY,X
A58D CA                         DEX
A58E 10 F7                      BPL     BBLP3
                        ;                               A=MSB WHICH SHOULD BE NEG.
A590 4C A8 9C                   JMP     BINFP2          ;CONVERT TO NEW FLOATING # (SHOULD BE NEG.) AND RETURN

A593                SNUM25
A593 48                         PHA
A594 A5 87                      LDA     DHOFLG
A596 20 83 AB                   JSR     INTMUL
A599 68                         PLA
```

```
A59A 38                        SEC
A59B E9 30                     SBC     #'0                 ;'0-'9 -> 0-9
A59D C9 11                     CMP     #'A-'0
A59F 90 02                     BCC     SNUM30
A5A1 E9 07                     SBC     #'A-'0-10           ;A-F -> 10-15
A5A3                   SNUM30
A5A3 20 A1 AB                  JSR     INTADD
A5A6 4C 4E A5                  JMP     SNUM20
A5A9                   SNUM40
A5A9 A9 00                     LDA     #TOKBUF             ;CONVERT ASCII TO FP IN REGO
A5AB A0 05                     LDY     #TOKBUF/256
A5AD 85 F3                     STA     INBUFF
A5AF 84 F4                     STY     INBUFF+1
A5B1 A9 00                     LDA     #0
A5B3 85 F2                     STA     CIX
A5B5 20 00 D8                  JSR     AFP

A5B8                   SNUM50
A5B8 60                        RTS
```

```
                        ;                       ROUTINES CORRESPONDING TO KEYWORDS

A5B9                    SABSVA
A5B9 A5 D4                      LDA     FR0             ;FR0 <- ABSVAL(FR0)
A5BB 29 7F                      AND     #$7F
A5BD 85 D4                      STA     FR0
A5BF                    RETURN
A5BF 60                         RTS
A5C0                    SACOS                           ;ARCCOS(FR0) = ARCTAN(SQRT(1-FR0*FR0)/FR0)
A5C0 A5 D4                      LDA     FR0
A5C2 D0 06                      BNE     SAC30
                        ;                               ARCCOS(0) = 90 DEG = PI/2 RAD.  SPECIAL CASE BECAUSE TAN UNDEFINED
A5C4                    SAC10
A5C4 20 09 A5                   JSR     PIOVL           ;LOAD X & Y REGS TO GET PI/2, 90 OR 100
A5C7 4C 89 DD                   JMP     FLDOR

A5CA                    SAC30
A5CA 48                         PHA
A5CB 20 9D A4                   JSR     ARGSUB          ;FR1 <- SQRT(1-FR0*FR0)
A5CE A5 E0                      LDA     FR1
A5D0 D0 12                      BNE     SAC34
A5D2 68                         PLA                     ;ABSVAL(FR0) = 1
A5D3 30 03                      BMI     S180PI
A5D5 4C 9E A2                   JMP     PCLR0           ;FR0=+1.  ARCCOS(+1) = 0
A5D8                    S180PI                          ;FR0 <- 180 OR 200 OR PI, DEPENDING ON RADFLG
A5D8 A6 FB                      LDX     RADFLG
A5DA D0 03                      BNE     SAC31
A5DC 4C 3C A9                   JMP     SPI             ;RAD => PI
A5DF                    SAC31
A5DF A9 B4                      LDA     #180            ;DEG => 180
                        ;       CPX     #GRADON
                        ;       BNE     SAC32
                        ;       LDA     #200            ;GRAD => 200
                        ;SAC32
A5E1 4C A7 A2                   JMP     PSET0

A5E4                    SAC34
A5E4 20 9B AB                   JSR     SFDIV
A5E7 20 03 AA                   JSR     SRECIP
A5EA 20 4C B4                   JSR     SATAN
A5ED 68                         PLA
A5EE 10 CF                      BPL     RETURN
A5F0 20 B6 DD                   JSR     FMOVE           ;COS <0 => ADD 180 DEG OR 200 GRAD OR PI TO ARCCOS
A5F3 20 D8 A5                   JSR     S180PI
A5F6 4C AA AB                   JMP     SFADD
                                .IF     ASMBL
                        SACOSH                          ;ARCCOSH(X) = LN(X+SQRT(SQUARE(X)-1))
                                JSR     FPUSH0
                                JSR     SSQUAR          ;X*X
                                LDA     #1
                                JSR     INTSUB          ;1-X*X
                                JSR     SCHGSG          ;X*X-1
                        AHYPSB                          ;FR0 <- LN(TOS + SQRT(FR0))
                                JSR     SSQRT           ;SQRT(X*X-1)
                                JSR     FPOP1
                                JSR     SFADD           ;X+SQRT(X*X-1)
                                JMP     SLN             ;LN(X+SQRT(X*X-1))
                                .ENDIF
```

```
A5F9                    SADV                            ;PUT CR ON PRINTER
A5F9 A6 9D                      LDX     PRNFLG          ;PRINTER ON ALREADY?
A5FB D0 0B                      BNE     SADV10          ;YES
A5FD 20 E8 A8                   JSR     SON             ;NO.  TURN ON
A600 B0 BD                      BCS     RETURN          ;ERROR
A602 20 08 A6                   JSR     SADV10          ;OUTPUT CR
A605 4C C3 A8                   JMP     SOFF            ;TURN OFF & RETURN
A608                    SADV10                          OUTPUT CR & RETURN
                        ;
A608 A5 CA                      LDA     DSPFLG
A60A 48                         PHA                     ;ALWAYS PRINT
A60B A9 00                      LDA     #0
A60D 85 CA                      STA     DSPFLG
A60F 20 85 A3                   JSR     PUTCRP
A612 68                         PLA
A613 85 CA                      STA     DSPFLG          ;RESTORE FLAG
A615 60                         RTS
A616                    SALG
A616 A9 01                      LDA     #ALGP
A618 D0 02                      BNE     SALG10          ;JMP
A61A                    SALGN
A61A A9 02                      LDA     #ALGNOP
A61C                    SALG10
A61C 4C 3F AA                   JMP     SRPN10
A61F                    SAND                            ;X <- X AND Y
A61F A9 00                      LDA     #0
A621 4C 2F AB                   JMP     DOLOP
A624                    SASIN                           ;ARCSIN(FR0) = ARCTAN(FR0/SQRT(FR0*FR0))
A624 20 9D A4                   JSR     ARCSUB          ;FR1 <- SQRT(1-FR0*FR0)
A627 A5 E0                      LDA     FR1
A629 D0 0C                      BNE     SAS10
A62B A5 D4                      LDA     FR0             ;|FR0| = 1
A62D 08                         PHP
A62E 20 C4 A5                   JSR     SAC10           ;FR0 <- 90 OR PI/2
A631 28                         PLP
A632 10 28                      BPL     BIN100          ;RETURN  FR0 = +1.  ARCSIN(+1) = 90 OR PI/2
A634 4C FE A6                   JMP     SCHGSG          ;FR0 = -1.  ARCSIN(-1) = -90 OR -PI/2
A637                    SAS10
A637 20 9B AB                   JSR     SFDIV
A63A 4C 4C B4                   JMP     SATAN
                                .IF     ASMBL
                        SASINH                          ;ARCSINH(X) = SIGN(X) * LN(ABSVAL(X)+SQRT(SQUARE(X)+1))
                                LDA     FR0
                                PHA                     ;SAVE SIGN
                                JSR     SABSVA          ;ABSVAL(X)
                                JSR     FPUSH0
                                JSR     SSQUAR          ;X*X
                                LDA     #1
                                JSR     INTADD          ;X*X+1
                                JSR     AHYPSB          ;LN(X+SQRT(X*X+1))
                                PLA
                                BPL     BIN100          ;RETURN
                                JMP     SCHGSG          ;IF SIGN IS NEGATIVE THEN ARCSINH(X) < 0
                        SATANH                          ;ARCTANH(X) = (LN((1+X)/(1-X)))/2
                                JSR     FPUSH0
                                LDA     #1
                                JSR     INTSUB          ;1-X
                                JSR     SXCHGY
                                LDA     #1
                                JSR     INTADD          ;1+X
```

```
                                JSR     FPOP1
                                JSR     SFDIV           ;(1+X)/(1-X)
                                JSR     SLN             ;LN((1+X)/(1-X))
                        DIVTWO                          ;MULTIPLY BY 1/2   (DIVIDE BY 2)
                                LDX     #FHALF
                                LDY     #FHALF/256
                                JSR     FLD1R
                                JMP     SFMUL
                                .ENDIF
A63D                    SBIN                            ;BASE 2 OR BINARY
A63D A9 02                      LDA     #2
A63F 20 BC A8                   JSR     SOCT10          ;CHANGE DHOFLG, STATUS MESSAGE
A642 A5 A7                      LDA     BITINT
A644 C9 11                      CMP     #17
A646 90 14                      BCC     BIN100
A648 A9 A8                      LDA     #BIMSG
A64A 20 E5 9B                   JSR     ERRSUB          ;BINARY REQUIRES 16 BITS OR LESS
A64D A9 31                      LDA     #'1
A64F 8D 0E 05                   STA     TOKBUF+NUMLEN-2
A652 A9 36                      LDA     #'6
A654 8D 0F 05                   STA     TOKBUF+NUMLEN-1
A657 A9 10                      LDA     #16
A659 20 7D A6                   JSR     SBITS2
A65C                    BIN100
A65C 60                         RTS
A65D                    SBITS                           ;SET OCTAL, HEX WORD LENGTH TO 1-32 BITS
                        ;                               BINARY TO 1-16 BITS
A65D A9 39                      LDA     #BTSMSG         ;DISPLAY "ENTER 1-32"
A65F 20 6A 9C                   JSR     PUTMSG
A662 20 9E A1                   JSR     GETINT          ;GET INTEGER
A665 D0 11                      BNE     SBERR           ;NOT GOOD - ERROR ALREADY REPORTED
A667 AA                         TAX
A668 F0 0E                      BEQ     SBERR           ;TOO SMALL
A66A A4 87                      LDY     DHOFLG
A66C C0 02                      CPY     #2
A66E D0 04                      BNE     SBITO
A670 C9 11                      CMP     #16+1
A672 B0 04                      BCS     SBERR
A674                    SBITO
A674 C9 21                      CMP     #32+1
A676 90 05                      BCC     SBITS2          ;TOO LARGE?
A678                    SBERR
A678 A9 76                      LDA     #BITMSG
A67A 4C E5 9B                   JMP     ERRSUB
A67D                    SBITS2
A67D 85 A7                      STA     BITINT
A67F AA                         TAX
A680 CA                         DEX
A681 8A                         TXA
A682 A2 00                      LDX     #0
A684 86 C0                      STX     BINARY+3
A686 86 BF                      STX     BINARY+2
A688 86 BE                      STX     BINARY+1
A68A 86 BD                      STX     BINARY+0        ;SET BINARY TO 0
A68C 38                         SEC
A68D 20 30 A5                   JSR     SLSHF2          ;SHIFT LEFT BITINT BITS WITH CARRY
A690 A2 03                      LDX     #3
A692                    SBLP1
A692 B5 BD                      LDA     BINARY,X
A694 95 B1                      STA     BITBIN,X
```

```
A696 CA                     DEX
A697 10 F9                  BPL     SBLP1
A699 20 12 A5               JSR     SCMP2           ;TAKE COMP
A69C A2 03                  LDX     #3
A69E                SBLP2
A69E B5 BD                  LDA     BINARY,X
A6A0 95 B9                  STA     BINMIN,X
A6A2 CA                     DEX
A6A3 10 F9                  BPL     SBLP2
A6A5 20 12 A5               JSR     SCMP2
A6A8 A9 01                  LDA     #1
A6AA 38                     SEC
A6AB 20 30 A5               JSR     SLSHF2          ;(2^BITINT)-1
A6AE A2 03                  LDX     #3
A6B0                SBLP3
A6B0 B5 BD                  LDA     BINARY,X
A6B2 95 B5                  STA     BITBN2,X
A6B4 CA                     DEX
A6B5 10 F9                  BPL     SBLP3

A6B7 A2 18                  LDX     #DBITS
A6B9                DSTAT
A6B9 86 55                  STX     COLCRS
A6BB A2 01                  LDX     #ROWSTT
A6BD 86 54                  STX     ROWCRS
A6BF A5 CA                  LDA     DSPFLG          ;ALWAYS DISPLAY
A6C1 48                     PHA
A6C2 A9 00                  LDA     #0
A6C4 85 CA                  STA     DSPFLG
A6C6 A2 30                  LDX     #'0             ;CONVERT INT 0-99 TO CHAR 00-99
A6C8 8E 00 05               STX     TOKBUF
A6CB A5 A7                  LDA     BITINT
A6CD                DSTLP
A6CD C9 0A                  CMP     #10
A6CF 90 07                  BCC     DST10
A6D1 E9 0A                  SBC     #10             ;(CARRY SET)
A6D3 EE 00 05               INC     TOKBUF
A6D6 D0 F5                  BNE     DSTLP           ;JMP
A6D8                DST10
A6D8 69 30                  ADC     #'0             ;(CARRY CLEAR)
A6DA 8D 01 05               STA     TOKBUF+1        ;LSDIGIT
A6DD A2 02                  LDX     #2              ;2 CHARS
A6DF A9 00                  LDA     #TOKBUF         ;LSB OF ADDR
A6E1 A0 05                  LDY     #TOKBUF/256
A6E3 20 5E A3               JSR     PUTCHS
A6E6 68                     PLA
A6E7 85 CA                  STA     DSPFLG          ;RESTORE OLD DSPFLG
A6E9 60                     RTS

A6EA                SCELSI
A6EA A9 5A                  LDA     #CELMSG         ;DISPLAY "-> FAHRENHEIT"
A6EC 20 6A 9C               JSR     PUTMSG
A6EF A2 3C                  LDX     #C1PT8          ;CELSIUS -> FAHRENHEIT F=(9/5)*C+32
A6F1 A0 B6                  LDY     #C1PT8/256
A6F3 20 98 DD               JSR     FLD1R
A6F6 20 8C AB               JSR     SFMUL
A6F9 A9 20                  LDA     #32
A6FB 4C A1 AB               JMP     INTADD
A6FE                SCHGSG
```

```
A6FE A5 D4                LDA     FR0             ;FR0<- -FR0
A700 F0 04                BEQ     SCH10
A702 49 80                EOR     #$80
A704 85 D4                STA     FR0
A706                SCH10
A706 60                   RTS
A707                SCLINI                        ;CLEAR MEM(3)-MEM(9) FOR INTEREST/STAT
A707 A2 29                LDX     #FPREC*7-1
A709 A9 00                LDA     #0
A70B 9D 12 08       CLLP  STA     FPREC*3+MEMORY,X
A70E CA                   DEX
A70F 10 FA                BPL     CLLP
A711 4C 14 9D             JMP     DMEMAL          ;CHANGE MEM DISPLAY
A714                SCLMEM                        ;CLEAR MEMORY
A714 A0 08                LDY     #MEMORY/256
A716 A2 0B                LDX     #MEMORY/256+3
A718 20 C4 A3             JSR     RAMCLR
A71B                DMCLR                         ;DISPLAY 0'S IN MEM 0-9
A71B 20 9C A0             JSR     FPUSH0          ;SAVE X
A71E 20 9E A2             JSR     PCLR0           ;AND DISPLAY 0 IN CURRENT FIX
A721 20 A0 9E             JSR     TOKNUM
A724 A9 09                LDA     #9
A726 85 AF                STA     MEMNUM
A728                DMCLP
A728 20 2E 9D             JSR     DSPMEM
A72B C6 AF                DEC     MEMNUM
A72D 10 F9                BPL     DMCLP
A72F 4C 7E A0             JMP     FPOP0           ;RELOAD X
A732                SCLR
A732 20 9E A2             JSR     PCLR0           ;CLEAR X, STACK
A735 4C 46 AA             JMP     SCLSTK
A29E                SCLX  =       PCLR0
                          .IF     ASMBL
                    SCOMBI                        ;Y COMBI X = C(N,R) = N!/(R!(N-R)!) = P(N,R)/R! =   (Y PERMU X)/X!
                          JSR     FPUSH0
                          JSR     SFACT0
                          LDX     #FTEMP2
                          LDY     #FTEMP2/256
                          JSR     FSTOR
                          JSR     FPOP0
                          JSR     SPERMU
                          LDX     #FTEMP2
                          LDY     #FTEMP2/256
                          JSR     FLD1R           ;RELOAD X!
                          JSR     SFDIV
                          JMP     SROUND
                          .ENDIF
A738                SCOMPL                        ;FR0 <- COMPLEMENT(FR0)
A738 20 15 9E             JSR     FPBIN
A73B 20 12 A5             JSR     SCMP2
A73E 4C A3 9C             JMP     BINFP
                          .IF     ASMBL           ;ASSEMBLE ONLY IF ASMBL SET
                    SCOSH                         ;COSH(X) <- (EXPE(X)+EXPE(-X))/2
                          JSR     HYPSUB          ;FR0 <- EXPE(X), FR1 <- EXPE(-X)
                          JSR     SFADD           ;EXPE(X) + EXPE(-X)
                          JMP     DIVTWO          ;DIVIDE BY 2 AND RETURN

                    SDAY
                          JSR     DAYSUB          ;MMDD.YYYY -> FACTOR
                          BCS     SCH10           ;ERROR => RETURN IMMEDIATELY
```

```
                        JSR     CLNUM           ;CLEAR TOKBUF
                ;                               ;FACTOR <- FACTOR MOD 7
                        LDA     #7
                        JSR     INTMOD
                        LDA     FRO+1           ;0-6
                        ASL     A
                        ADC     FRO+1           ;3*(FACTOR MOD 7)
                        TAX                     ;MOVE DAY OF WEEK CHARS TO END OF TOKBUF
                        LDY     #NUMLEN-3
                SDAYLP
                        LDA     DAYTBL,X
                        STA     TOKBUF,Y
                        INX
                        INY
                        CPY     #NUMLEN
                        BNE     SDAYLP
                        JSR     DAYDSP          ;DISPLAY DAY OF WEEK IN NUMBER LOC
                        JMP     DAYCOM          ;DISPLAY "***"
                SDBD                            ;DAYS BETWEEN DATES
                        JSR     DAYSUB          ;Y DBD X = DAYSUB(Y) - DAYSUB(X)
                        BCS     SFDON           ;ERROR => RETURN
                        JSR     SXCHGY
                        JSR     DAYSUB          ;DAYSUB Y
                        BCS     SFDON           ;ERROR => RETURN
                        JSR     FPOP1
                        JMP     SFSUB
                        .ENDIF
A741                    SDEC                    ;SET DECIMAL MODE
A741 A9 00              LDA     #0              ;DECIMAL MODE
A743 4C BC A8           JMP     SOCT10
A746                    SDEG                    ;SET DEGREE MODE
A746 A9 06              LDA     #6              ;DEGON
A748 4C DE A9           JMP     SRAD10
A74B                    SDIV
A74B 20 E7 A4           JSR     MEMSUB          ;MEM <- MEM/X
A74E 20 9B AB           JSR     SFDIV
A751 4C E1 AA           JMP     SSUM10
A754                    SDMS                    ;DMS -> DECIMAL DEGREES DD.MMSSSSS -> DD.DDDD
A754 A9 91              LDA     #ZDMS
A756 A2 64              LDX     #100            ;NUMERATOR, MOD BASE
A758 A0 3C              LDY     #60             ;DENOMINATOR
A75A D0 06              BNE     DEGSUB          ;JMP
A75C                    SDCDEG                  ;DECIMAL DEG -> DMS
A75C A9 9E              LDA     #ZDCDEG
A75E A2 3C              LDX     #60
A760 A0 64              LDY     #100
A762                    DEGSUB
A762 8E 65 0B           STX     XSAVE2
A765 8C 66 0B           STY     YSAVE2
A768 20 6A 9C           JSR     PUTMSG          ;DISPLAY MESSAGE -> DMS OR -> DECIMAL DEGREES
                ;       INT(X) + (FRACT(X) MODFAC (1/XSAVE2))/YSAVE2 + (FRACT(X) MOD (1/XSAVE2))*XSAVE2^2/YSAVE2^2
A76B 20 B6 DD           JSR     FMOVE
A76E 20 55 A8           JSR     SINTEG
A771 20 9C A0           JSR     FPUSH0          ;SAVE INT(X)
A774 20 6F A0           JSR     FMOVE2
A777 20 44 A8           JSR     SFRACT
A77A 20 9C A0           JSR     FPUSH0          ;SAVE FRACT(X)
A77D AD 65 0B           LDA     XSAVE2
A780 20 A7 A2           JSR     PSET0           ;INT -> FP
A783 20 03 AA           JSR     SRECIP
```

```
A786 20 95 A8           JSR     SMOD            ;FRACT(X) MOD (1/XSAVE2)  ALSO SETS UP MODFAC
A789 20 9C A0           JSR     FPUSHO          ;SAVE MOD
A78C AD 65 0B           LDA     XSAVE2
A78F 20 A7 A2           JSR     PSETO
A792 20 AB AA           JSR     SSQUAR
A795 20 9C A0           JSR     FPUSHO
A798 AD 66 0B           LDA     YSAVE2
A79B 20 A7 A2           JSR     PSETO
A79E 20 AB AA           JSR     SSQUAR
A7A1 20 98 AB           JSR     SPDIV           ;XSAVE2^2/YSAVE2^2
A7A4 20 89 AB           JSR     SPMUL           ;MULTIPLY BY MOD
A7A7 20 A7 AB           JSR     SPADD           ;ADD INT(X)
A7AA 20 9C A0           JSR     FPUSHO          ;SAVE RESULT ON STACK
A7AD AD 66 0B           LDA     YSAVE2
A7B0 20 A7 A2           JSR     PSETO
A7B3 20 B6 DD           JSR     FMOVE
A7B6 20 34 A0           JSR     FLDOM           ;LOAD MODFAC
A7B9 20 9B AB           JSR     SFDIV           ;MODFAC/YSAVE2
A7BC 4C A7 AB           JMP     SPADD           ;ADD TO PREVIOUS RESULT & RETURN

A7BF            SEXPE                           ;FRO <- E^FRO
A7BF 20 C0 DD           JSR     EXP
A7C2 4C BC AB           JMP     CRYCHK
A7C5            SEXPTE                          ;FRO <- 10^FRO
A7C5 20 CC DD           JSR     EXP10
A7C8 4C BC AB           JMP     CRYCHK
A7CB            SFACTO                          ;X! = X(X-1)(X-2).....
A7CB 20 55 A0           JSR     FSTOT           ;GINT2 WILL LOAD
A7CE 20 BD A1           JSR     GINT2           ;A= INTEGER 0-255
A7D1 D0 21              BNE     SFERR           ;ERROR
A7D3 C9 45              CMP     #69
A7D5 B0 1D              BCS     SFERR           ;ERROR - TOO LARGE
A7D7 AA                 TAX
A7D8 D0 02              BNE     SF10
A7DA E8                 INX
A7DB 8A                 TXA                     ;0! = 1! = 1
A7DC            SF10
A7DC 48                 PHA
A7DD 20 A7 A2           JSR     PSETO           ;FACT <- N
A7E0            SFLP
A7E0 20 B6 DD           JSR     FMOVE
A7E3 68                 PLA
A7E4 C9 03              CMP     #3
A7E6 90 11              BCC     SFDON
A7E8 E9 01              SBC     #1              ;X <- X-1
A7EA 48                 PHA
A7EB 20 A7 A2           JSR     PSETO           ;INT -> FP
A7EE 20 DB DA           JSR     FMUL            ;FACT <- FACT * X
A7F1 90 ED              BCC     SFLP
A7F3 68                 PLA                     ;CARRY SET => MULTIPLY ERROR (SHOULDN'T HAPPEN)
A7F4            SFERR
A7F4 A9 76              LDA     #BITMSG
A7F6 4C E5 9B           JMP     ERRSUB
A7F9            SFDON
A7F9 60                 RTS
A7FA            SFAHRE
A7FA A9 66              LDA     #FAHMSG         ;DISPLAY "->CELSIUS"
A7FC 20 6A 9C           JSR     PUTMSG
A7FF A9 20              LDA     #32
A801 20 B0 AB           JSR     INTSUB
```

```
A801 20 B0 AB              JSR     INTSUB
```

COLLEEN CALCULATOR, BY C SHAW

```
A804 20 FE A6              JSR     SCHGSG
A807 A2 3C                 LDX     #C1PT8          ;FAHRENHEIT -> CELSIUS C=(5/9)*(F-32)
A809 A0 B6                 LDY     #C1PT8/256
A80B 20 98 DD              JSR     FLD1R
A80E 4C 9B AB              JMP     SFDIV
A811                SFIX
A811 A9 1D                 LDA     #FIXMSG         ;DISPLAY "ENTER 0-7 OR 9"
A813 20 6A 9C              JSR     PUTMSG
A816 20 9E A1              JSR     GETINT          ;GET NEXT TOKEN: WANT INTEGER 0-9
A819 D0 24                 BNE     SFXERR          ;ERROR
A81B C9 08                 CMP     #8
A81D F0 20                 BEQ     SFXERR          ;DON'T ALLOW FIX 8
A81F C9 0A                 CMP     #9+1
A821 B0 1C                 BCS     SFXERR          ;ERROR - TOO LARGE
A823                SFIX2                          ;ENTRY PT.  IF A=VALID INTEGER
A823 85 CE                 STA     FIXNUM
A825 69 30                 ADC     #'0             ;-> ASCII
A827 AA                    TAX
A828 A9 1E                 LDA     #DFIX           ;DISPLAY IN STATUS AREA OF SCREEN
A82A 85 55                 STA     COLCRS
A82C A9 01                 LDA     #ROWSTT
A82E 85 54                 STA     ROWCRS
A830 A5 CA                 LDA     DSPFLG
A832 48                    PHA
A833 A9 00                 LDA     #0
A835 85 CA                 STA     DSPFLG
A837 8A                    TXA
A838 20 F2 A2              JSR     PTCHR
A83B 68                    PLA
A83C 85 CA                 STA     DSPFLG
A83E 60                    RTS
A83F                SFXERR
A83F A9 76                 LDA     #BITMSG         ;NUMBER OUT OF RANGE
A841 4C E5 9B              JMP     ERRSUB
A844                SFRACT                         ;FR0 <- FRACTIONAL PART OF FR0
A844 20 B6 DD              JSR     FMOVE
A847 20 55 A8              JSR     SINTEG          ;-(INT(FR0)-FR0)
A84A 20 B9 AB              JSR     SFSUB
A84D 4C FE A6              JMP     SCHGSG
                           .IF     ASMBL
                    SGRAD                          ;SET GRAD MODE
                           LDA     #GRADON
                           JMP     SRAD10
                           .ENDIF
A850                SHEX                           ;SET HEX MODE
A850 A9 10                 LDA     #16
A852 4C BC A8              JMP     SOCT10
A855                SINTEG                         ;FR0 <- INT(FR0)
A855 A5 D4                 LDA     FR0
A857 F0 1B                 BEQ     SINTRT          ;0 IS OK
A859 29 7F                 AND     #$7F            ;COMPUTE DECIMAL PT LOC
A85B 38                    SEC
A85C E9 40                 SBC     #$40
A85E 10 04                 BPL     SINT10
A860 A9 FE                 LDA     #-2
A862 D0 04                 BNE     SINT30          ;JMP
A864                SINT10
A864 C9 04                 CMP     #FPREC-2
A866 B0 0C                 BCS     SINTRT          ;LARGE => NO FRACTIONAL PART, DON'T DO ANYTHING
A868                SINT30
```

```
A868 AA                     TAX
A869 E8                     INX
A86A E8                     INX
A86B A9 00                  LDA     #0
A86D                SINT40
A86D 95 D4                  STA     FR0,X
A86F E8                     INX
A870 E0 06                  CPX     #FPREC
A872 90 F9                  BCC     SINT40
A874                SINTRT
A874 60                     RTS
                            ;<0 => ALL FRACTIONAL (ALL 0)
A875                SINTME                  ;SET FLAG TO DISPLAY INTERMEDIATE RESULTS IN ALG
A875 A5 A0                  LDA     INTERM
A877 49 01                  EOR     #1
A879 85 A0                  STA     INTERM
A87B 60                     RTS
A87C                SLN                     ;FR0 <- LN(FR0)
A87C 20 CD DE               JSR     LOG
A87F 4C BC AB               JMP     CRYCHK
A882                SLOGTE                  ;FR0 <- LOG10(FR0)
A882 20 D1 DE               JSR     LOG10
A885 4C BC AB               JMP     CRYCHK
A888                SLSHF                   ;Y LSHF X
A888 A9 00                  LDA     #0
A88A 4C 60 AA               JMP     SHFSUB
A88D                INTMOD                  ;FR0 <- FR0 MOD A  (ALSO MODFAC <- INT(Y/X))
A88D 48                     PHA
A88E 20 9C A0               JSR     FPUSH0
A891 68                     PLA
A892 20 A7 A2               JSR     PSET0
A895                SMOD                    ;Y MOD X = Y-X*INT(Y/X)
A895 20 B6 DD               JSR     FMOVE
A898 20 3B A0               JSR     FLD0S
A89B 20 AA A0               JSR     FPUSH1
A89E 20 9B AB               JSR     SFDIV       ;Y/X
A8A1 20 55 A8               JSR     SINTEG      ;INT(Y/X)
A8A4 A2 4C                  LDX     #MODFAC
A8A6 A0 0B                  LDY     #MODFAC/256
A8A8 20 A7 DD               JSR     FSTOR       ;SAVE INT(Y/X) IN MODFAC
A8AB 20 75 A0               JSR     FPOP1       ;X

A8AE 20 8C AB               JSR     SFMUL       ;INT(Y/X)*X
A8B1 20 B6 DD               JSR     FMOVE
A8B4 20 7E A0               JSR     FPOP0       ;Y
A8B7 4C B9 AB               JMP     SFSUB       ;Y - INT(Y/X)*X
```

```
A8BA                    SOCT
A8BA A9 08                      LDA     #8              ;SET OCTAL MODE
A8BC                    SOCT10
A8BC 85 87                      STA     DHOFLG
A8BE A9 10                      LDA     #DDEC
A8C0 4C 0D A3                   JMP     CHSTAT          ;CHANGE STATUS LINE ON SCREEN
A8C3                    SOFF                            ;CLOSE PRINTER
A8C3 A9 4F                      LDA     #'O
A8C5 8D 00 05                   STA     TOKBUF
A8C8 A9 46                      LDA     #'F
A8CA 8D 01 05                   STA     TOKBUF+1
A8CD 8D 02 05                   STA     TOKBUF+2
A8D0 A9 03                      LDA     #3
A8D2 85 82                      STA     TOKPTR
A8D4 A9 20                      LDA     #DOFF
A8D6 20 0D A3                   JSR     CHSTAT          ;CHANGE STATUS LINE ON SCREEN
A8D9 A2 20                      LDX     #PIOCB
A8DB A9 0C                      LDA     #CLOSE
A8DD 9D 42 03                   STA     ICCOM,X
A8E0 20 56 E4                   JSR     CIOV
A8E3 A2 00                      LDX     #0
A8E5 86 9D                      STX     PRNFLG          ;ALWAYS OFF
A8E7 60                         RTS
A8E8                    SON                             ;OPEN PRINTER FOR OUTPUT
A8E8 A6 9D                      LDX     PRNFLG
A8EA D0 3C                      BNE     POPN20          ;ALREADY OPEN
A8EC A2 20                      LDX     #PIOCB
A8EE A9 03                      LDA     #OPEN
A8F0 9D 42 03                   STA     ICCOM,X
A8F3 A9 BB                      LDA     #PBUFF
A8F5 9D 44 03                   STA     ICBAL,X
A8F8 A9 B6                      LDA     #PBUFF/256
A8FA 9D 45 03                   STA     ICBAH,X
A8FD A9 08                      LDA     #OUTPUT
A8FF 9D 4A 03                   STA     ICAX1,X
A902 20 56 E4                   JSR     CIOV
A905 C0 01                      CPY     #SUCCES
A907 F0 08                      BEQ     POPN10
A909                    OFFERR
A909 20 C3 A8                   JSR     SOFF
A90C A9 8E                      LDA     #NOPMSG
A90E 4C E5 9B                   JMP     ERRSUB
A911                    POPN10
A911 A2 01                      LDX     #1
A913 86 9D                      STX     PRNFLG
A915 A9 4F                      LDA     #'O
A917 8D 00 05                   STA     TOKBUF
A91A A9 4E                      LDA     #'N
A91C 8D 01 05                   STA     TOKBUF+1
A91F A9 02                      LDA     #2
A921 85 82                      STA     TOKPTR
A923 A9 20                      LDA     #DOFF
A925 20 0D A3                   JSR     CHSTAT
A928                    POPN20
A928 18                         CLC                     ;NO ERROR
A929 60                         RTS
A92A                    SOR                             ;Y OR X
A92A A9 01                      LDA     #1
A92C 4C 2F AB                   JMP     DOLOP
A92F                    SPERCE
```

```
A92F 20 7B A0                JSR     FMVPOP          ;Y%X = (X*Y)/100        (ORDERING IS UNIMPORTANT)
A932 20 8C AB                JSR     SFMUL
A935 A5 D4                   LDA     FR0
A937 F0 02                   BEQ     SPE10
A939 C6 D4                   DEC     FR0
A93B                 SPE10
A93B 60                      RTS
                             .IF     ASMBL
                     SPERMU
                             JSR     FMOVE           ;Y PERMU X = P(N,R) = N!/(N-R)! = Y!/(Y-X)!
                             JSR     FLDOS
                             JSR     SFSUB           ;Y-X
                             JSR     SFACTO          ;(Y-X)!
                             JSR     SXCHGY          ;LOAD Y
                             JSR     SFACTO          ;Y!
                             JSR     FPOP1           ;LOAD (Y-X)!
                             JSR     SFDIV
                             JMP     SROUND
                             .ENDIF
A93C                 SPI                             ;X <- PI
A93C A5 99                   LDA     RPNALG
A93E D0 03                   BNE     SPI10
A940 20 9C A0                JSR     FPUSH0          ;PUSH OLD X IF RPN
A943                 SPI10
A943 A2 00                   LDX     #PICONST
A945 A0 B6                   LDY     #PICONST/256
A947 4C 89 DD                JMP     FLDOR
A94A                 SPOLAR  ;->RECT Y=THETA X=R   NEW Y=R*SIN(THETA)  NEW X=SQRT(SQU(R)-SQU(Y))
A94A A9 C3                   LDA     #ZPOLAR
A94C 20 6A 9C                JSR     PUTMSG          ;DISPLAY "->RECTANGULAR"
A94F 20 55 A0                JSR     FSTOT           ;SAVE X=R
A952 20 7E A0                JSR     FPOP0           ;LOAD THETA=Y
A955 20 AA B3                JSR     SSIN            ;SIN(THETA)
A958 20 4E A0                JSR     FLD1T           ;RELOAD R
A95B 20 8C AB                JSR     SFMUL           ;Y = R*SIN(THETA)
A95E 20 9C A0                JSR     FPUSH0          ;SAVE NEW Y ON STACK
A961 20 AB AA                JSR     SSQUAR          ;Y*Y
A964 20 9C A0                JSR     FPUSH0          ;SAVE Y*Y
A967 20 47 A0                JSR     FLDOT           ;LOAD R
A96A 20 AB AA                JSR     SSQUAR          ;R*R
A96D 20 75 A0                JSR     FPOP1           ;RELOAD Y*Y
A970 20 B9 AB                JSR     SFSUB           ;R*R - Y*Y
A973 4C BA B4                JMP     SSQRT           ;NEW X
A07E                 SPOP    =       FPOP0                   ;POP # OFF STACK IF RPN
A976                 SPOWER                          ;Y^X = EXP10(X* LOG10(Y))
A976 20 75 A0                JSR     FPOP1
A979 A5 E0                   LDA     FR1
A97B 10 09                   BPL     SPOW10
A97D 29 7F                   AND     #$7F
A97F 85 E0                   STA     FR1
A981 A9 76                   LDA     #BITMSG         ;Y<0 => ERRROR
A983 20 E5 9B                JSR     ERRSUB
A986                 SPOW10
A986 A5 D4                   LDA     FR0
A988 D0 05                   BNE     SPOW20
A98A A9 01                   LDA     #1              ;X=0 => RETURN(1)
A98C 4C A7 A2                JMP     PSET0
A98F                 SPOW20
A98F A5 E0                   LDA     FR1
A991 D0 0C                   BNE     SPOW40
```

```
A993 A5 D4              LDA     FRO             ;Y=0
A995 10 05              BPL     SPOW30
A997 A9 9B              LDA     #CRYMSG         ;X<0 => ERROR
A999 20 E5 9B           JSR     ERRSUB
A99C                SPOW30
A99C 4C 9E A2           JMP     PCLR0
A99F                SPOW40
A99F 20 55 A0           JSR     FSTOT
A9A2 20 AA A0           JSR     FPUSH1
A9A5 A9 01              LDA     #1
A9A7 85 AA              STA     INTFLG
A9A9 20 44 A8           JSR     SFRACT          ;TAKE FRACTIONAL PART
A9AC A5 D4              LDA     FR0
A9AE D0 0C              BNE     SPOW50
A9B0 20 3B A0           JSR     FLD0S           ;X IS INTEGER
A9B3 20 44 A8           JSR     SFRACT          ;TAKE FRACTIONAL PART
A9B6 A5 D4              LDA     FR0
A9B8 D0 02              BNE     SPOW50
A9BA C6 AA              DEC     INTFLG          ;Y IS INTEGER
A9BC                SPOW50
A9BC 20 7E A0           JSR     FPOP0           ;Y
A9BF 20 82 A8           JSR     SLOGTE
A9C2 20 4E A0           JSR     FLD1T           ;X
A9C5 20 8C AB           JSR     SFMUL
A9C8 20 C5 A7           JSR     SEXPTE
A9CB A5 AA              LDA     INTFLG
A9CD D0 03              BNE     SPOW60
A9CF 4C 51 AA           JMP     SROUND          ;BOTH INTEGER => ROUND TO INTEGER
A9D2                SPOW60
A9D2 60                 RTS
A9D3                SPRD
A9D3 20 E7 A4           JSR     MEMSUB          ;MEM <- MEM*X
A9D6 20 8C AB           JSR     SFMUL
A9D9 4C E1 AA           JMP     SSUM10
A09C                SPUSH   =       FPUSH0                  ;PUSH # ON STACK IF RPN
A9DC                SRAD                    ;SET RAD MODE
A9DC A9 00              LDA     #0              ;RADON
A9DE                SRAD10
A9DE 85 FB              STA     RADFLG
A9E0 A9 0B              LDA     #DDEG           ;CHANGE STATUS LINE ON SCREEN
A9E2 4C 0D A3           JMP     CHSTAT
A9E5                SRAND0                  ;X <- RANDOM NUMBER FROM 0 TO 65535
A9E5 A5 99              LDA     RPNALG
A9E7 D0 03              BNE     SRAN10
A9E9 20 9C A0           JSR     FPUSH0          ;IF RPN PUSH PREVIOUS # AS IN SPI
A9EC                SRAN10
A9EC AD 0A D2           LDA     RANDOM
A9EF 85 D4              STA     FR0
A9F1 A9 00              LDA     #0
A9F3 85 D5              STA     FR0+1
A9F5 4C AA D9           JMP     IFP
A9F8                SRCL
A9F8 20 E7 A4           JSR     MEMSUB          ;X <- MEM
A9FB                SRCL10
A9FB A5 99              LDA     RPNALG
A9FD F0 03              BEQ     SRCL20
A9FF 4C 75 A0           JMP     FPOP1           ;NOT RPN => POP X OFF STACK
AA02                SRCL20
AA02 60                 RTS
AA03                SRECIP
```

```
AA03 A9 01                LDA     #1
AA05 4C 92 AB             JMP     INTDIV          ;FRO <- 1/FRO
AA08              SRECTA  ;->POLAR NEW X=R=SQRT(SQU(X)+SQU(Y)) NEW Y=THETA=ASIN(Y/R)
AA08 A9 A5                LDA     #ZRECT
AA0A 20 6A 9C             JSR     PUTMSG          ;DISPLAY "-> POLAR"
AA0D 20 AB AA             JSR     SSQUAR          ;X*X
AA10 20 55 A0             JSR     FSTOT
AA13 20 3B A0             JSR     FLDOS
AA16 20 AB AA             JSR     SSQUAR          ;Y*Y
AA19 20 4E A0             JSR     FLD1T
AA1C 20 AA AB             JSR     SFADD
AA1F 20 BA B4             JSR     SSQRT           ;R = NEW X (TOS)
AA22 20 B6 DD             JSR     FMOVE
AA25 20 55 A0             JSR     FSTOT
AA28 20 7E A0             JSR     FPOPO           ;Y
AA2B 20 9B AB             JSR     SFDIV           ;Y/R
AA2E 20 24 A6             JSR     SASIN           ;THETA = NEW Y
AA31 20 9C A0             JSR     FPUSHO
AA34 4C 47 A0             JMP     FLDOT           ;LOAD NEW X
AA37              SROOT
AA37 20 03 AA             JSR     SRECIP          ;Y ROOT X = Y POWER 1/X
AA3A 4C 76 A9             JMP     SPOWER
AA3D              SRPN
AA3D A9 00                LDA     #0              ;RPN
AA3F              SRPN10
AA3F 85 99                STA     RPNALG
AA41 A9 06                LDA     #DALG           ;CHANGE STATUS LINE ON SCREEN
AA43 20 0D A3             JSR     CHSTAT
AA46              SCLSTK
AA46 A9 01                LDA     #1              ;CLEAR STACKS (LPAD ONLY)
AA48 85 A5                STA     OPPTR
AA4A A9 00                LDA     #0              ;X ONLY, NOTHING ON STACK
AA4C 85 A4                STA     FPPTR
AA4E 4C 51 9D             JMP     DSPSTK          ;DISPLAY STACK
AA51              SROUND                  ;ROUND(X) = INT(X+.5)
AA51 A2 6C                LDX     #FHALF
AA53 A0 DF                LDY     #FHALF/256
AA55 20 98 DD             JSR     FLD1R
AA58 20 AA AB             JSR     SFADD
AA5B 4C 55 A8             JMP     SINTEG
AA5E              SRSHF                   ;Y RSHF X
AA5E A9 01                LDA     #1
AA60              SHFSUB                  ;Y SHF X (RIGHT OR LEFT)
AA60 85 A8                STA     TO              ;1=>RIGHT, 0=> LEFT
AA62 A5 D4                LDA     FR0
AA64 10 0A                BPL     SHF05
AA66 29 7F                AND     #$7F            ;X<0: TAKE ABSOLUTE VALUE AND SHIFT IN OPPOSITE DIRECTION
AA68 85 D4                STA     FR0
AA6A A5 A8                LDA     TO
AA6C 49 01                EOR     #1
AA6E 85 A8                STA     TO
AA70              SHF05
AA70 20 D2 D9             JSR     FPI             ;FP -> INT
AA73 08                   PHP
AA74 20 B6 DD             JSR     FMOVE           ;FR1 <- FR0
AA77 20 7E A0             JSR     FPOPO           ;LOAD Y
AA7A 28                   PLP                     ;RELOAD CARRY FROM FPI
AA7B B0 0F                BCS     SHF10           ;ERROR => RETURN 0
AA7D A5 E1                LDA     FR1+1
AA7F D0 0B                BNE     SHF10           ;SHIFT > 256 => RETURN 0
```

```
AA7F D0 0B                      BNE     SHF10           ;SHIFT > 256 => RETURN 0
```

COLLEEN CALCULATOR, BY C SHAW

```
AA81 A5 E0                      LDA     FR1             ;LOAD LSB OF SHIFT
AA83 48                         PHA                     ;SAVE
AA84 20 15 9E                   JSR     FPBIN
AA87 68                         PLA
AA88 C5 A7                      CMP     BITINT
AA8A 90 03                      BCC     SHF15
AA8C                    SHF10
AA8C 4C 9E A2                   JMP     PCLR0           ;RETURN 0
AA8F                    SHF15
AA8F AA                         TAX
AA90 A5 A8                      LDA     T0
AA92 D0 08                      BNE     SHF20           ;RIGHT
AA94 18                         CLC                     ;LEFT
AA95 8A                         TXA
AA96 20 30 A5                   JSR     SLSHF2
AA99 4C A3 9C                   JMP     BINFP
AA9C                    SHF20
AA9C 46 BD                      LSR     BINARY+0        ;RIGHT
AA9E 66 BE                      ROR     BINARY+1
AAA0 66 BF                      ROR     BINARY+2
AAA2 66 C0                      ROR     BINARY+3
AAA4 CA                         DEX
AAA5 D0 F5                      BNE     SHF20
AAA7 4C A3 9C                   JMP     BINFP
AAAA                    SHF30
AAAA 60                         RTS
                                .IF     ASMBL
                        SSINH                           ;SINH(X) <- (EXPE(X) - EXPE(-X)) / 2
                                JSR     HYPSUB          ;FR0 <- EXPE(X), FR1 <- EXPE(-X)
                                JSR     SFSUB           ;EXPE(X) - EXPE(-X)
                                JMP     DIVTWO          ;DIVIDE BY 2 AND RETURN
                                .ENDIF
AAAB                    SSQUAR
AAAB 20 B6 DD                   JSR     FMOVE           ;X SQUARED = X*X
AAAE 4C 8C AB                   JMP     SFMUL
AAB1                    SSTO
AAB1 20 B1 A4                   JSR     GETMN           ;MEM <- X
AAB4 D0 1B                      BNE     SSTO20          ;ERROR
AAB6                    SSTO10
AAB6 20 D7 A4                   JSR     MEMLDR
AAB9 20 A7 DD                   JSR     FSTOR
AABC AD 6B 0B                   LDA     DMFLG           ;DISPLAY MEMORY INHIBIT?
AABF D0 10                      BNE     SSTO20          ;YES. DON'T DISPLAY, EVEN IN MEM DISPLAY AREA
AAC1 A5 CA                      LDA     DSPFLG          ;DISPLAY INHIBIT?
AAC3 F0 06                      BEQ     SSTO12          ;NO.
AAC5 20 A0 9E                   JSR     TOKNUM          ;YES. NO DISPLAY, JUST CONVERT FP TO ASCII
AAC8 4C CE AA                   JMP     SSTO14
AACB                    SSTO12
AACB 20 5D 9E                   JSR     FDSCOM          ;DISPLAY NUMBER IN SCROLL AREA WITH '***'
AACE                    SSTO14
AACE 4C 2E 9D                   JMP     DSPMEM          ;DISPLAY MEM IN MEMORY AREA ON SCREEN
AAD1                    SSTO20
AAD1 60                         RTS
AAD2                    SSUB
AAD2 20 E7 A4                   JSR     MEMSUB          ;MEM <- MEM-X
AAD5 20 B9 AB                   JSR     SFSUB
AAD8 4C E1 AA                   JMP     SSUM10
AADB                    SSUM
AADB 20 E7 A4                   JSR     MEMSUB          ;MEM <- MEM+X
AADE 20 AA AB                   JSR     SFADD
```

```
AAE1                    SSUM10
AAE1 20 B6 AA                   JSR     SSTO10
AAE4 4C 7E A0                   JMP     FPOPO           ;RELOAD X VALUE
AAE7                    STAN                            ;TAN(X) = SIN(X)/COS(X)
AAE7 20 9C A0                   JSR     FPUSHO
AAEA 20 9B B3                   JSR     SCOS
AAED 20 F9 AA                   JSR     SXCHGY
AAF0 20 AA B3                   JSR     SSIN
AAF3 20 75 A0                   JSR     FPOP1
AAF6 4C 9B AB                   JMP     SFDIV
                                .IF     ASMBL
                        STANH                           ;TANH(X) <- SINH(X)/COSH(X)
                                JSR     FPUSHO
                                JSR     SCOSH
                                JSR     SXCHGY
                                JSR     SSINH
                                JSR     FPOP1
                                JMP     SFDIV
                                .ENDIF
AAF9                    SXCHGY
AAF9 20 B6 DD                   JSR     FMOVE           ;FRO <-> TOS
AAFC 20 7E A0                   JSR     FPOPO
AAFF 4C AA A0                   JMP     FPUSH1
AB02                    SXCHM
AB02 20 E7 A4                   JSR     MEMSUB          ;X <==> MEM   (MEMSUB PUSHES FRO)
AB05 20 D7 A4                   JSR     MEMLDR          ;MEM <- X
AB08 20 60 A0                   JSR     FST1R
AB0B 20 F9 AA                   JSR     SXCHGY          ;EXCHANGE NEW X FOR OLD ON STACK
AB0E A5 AF                      LDA     MEMNUM
AB10 20 47 9D                   JSR     DSPM3           ;DISPLAY X IN MEM REG AREA
AB13 B0 15                      BCS     SX10            ;IF MEMNUM < 10
AB15 20 D7 A4                   JSR     MEMLDR
AB18 20 89 DD                   JSR     FLDOR
AB1B A5 CA                      LDA     DSPFLG
AB1D 48                         PHA
AB1E A9 00                      LDA     #0
AB20 85 CA                      STA     DSPFLG
AB22 A9 16                      LDA     #COLCMD+2
AB24 20 92 9E                   JSR     FDSP1
AB27 68                         PLA
AB28 85 CA                      STA     DSPFLG          ;RESTORE
AB2A                    SX10
AB2A 4C 7E A0                   JMP     FPOPO
AB2D                    SXOR                            ;X <- Y XOR X
AB2D A9 02                      LDA     #2
AB2F                    DOLOP                           ;X <- Y LOP X    0=>AND, 1=>OR, 2=>XOR
AB2F 85 A8                      STA     TO
AB31 20 15 9E                   JSR     FPBIN
AB34 A2 03                      LDX     #3
AB36                    LOPLP1
AB36 B5 BD                      LDA     BINARY,X
AB38 95 C1                      STA     BIN2,X
AB3A CA                         DEX
AB3B 10 F9                      BPL     LOPLP1
AB3D 20 7E A0                   JSR     FPOPO
AB40 20 15 9E                   JSR     FPBIN
AB43 A4 A8                      LDY     TO
AB45 A2 03                      LDX     #3
AB47                    LOPLP2
AB47 B5 BD                      LDA     BINARY,X
```

```
AB49 C0 00                    CPY     #0
AB4B D0 05                    BNE     LOP10
AB4D 35 C1                    AND     BIN2,X
AB4F 4C 5D AB                 JMP     LOP30
AB52                  LOP10
AB52 C0 01                    CPY     #1
AB54 D0 05                    BNE     LOP20
AB56 15 C1                    ORA     BIN2,X
AB58 4C 5D AB                 JMP     LOP30
AB5B                  LOP20
AB5B 55 C1                    EOR     BIN2,X
AB5D                  LOP30
AB5D 95 BD                    STA     BINARY,X
AB5F CA                       DEX
AB60 10 E5                    BPL     LOPLP2
AB62 4C A3 9C                 JMP     BINFP
AB65                  SXPRIM
AB65 60                       RTS
AB66                  SPRINT
                      ;                                       PRINT X REG
AB66 A5 9D                    LDA     PRNFLG                  ;PRINTER ON?
AB68 D0 0B                    BNE     SXR10                   ;YES.
AB6A 20 E8 AB                 JSR     SON                     ;NO. TURN ON
AB6D B0 13                    BCS     SXRTN                   ;ERROR
AB6F 20 75 AB                 JSR     SXR10                   ;DISPLAY & PRINT
AB72 4C C3 A8                 JMP     SOFF                    ;TURN OFF & RETURN
AB75                  SXR10
                      ;                                       DISPLAY & PRINT &RETURN
AB75 A5 CA                    LDA     DSPFLG                  ;ALWAYS PRINT
AB77 48                       PHA
AB78 A9 00                    LDA     #0
AB7A 85 CA                    STA     DSPFLG
AB7C 20 73 9E                 JSR     FDSPO                   ;PRINT NUMBER
AB7F 68                       PLA
AB80 85 CA                    STA     DSPFLG
AB82                  SXRTN
AB82 60                       RTS
```

COLLEEN CALCULATOR, BY C SHAW

```
AB83                    INTMUL                                  ;FR0 <- A*FR0
AB83 20 A2 A2                   JSR     LDINT                   ;FR1 <- FR0, FR0<-A
AB86 4C 8C AB                   JMP     SFMUL
AB89                    SPMUL
AB89 20 7B A0                   JSR     FMVPOP                  ;FR1 <- FR0 ; POP Y OFF STACK INTO FR0
AB8C                    SFMUL                                   ;X <- Y*X   FR0 <- FR0 * FR1
AB8C 20 DB DA                   JSR     FMUL
AB8F 4C BC AB                   JMP     CRYCHK
AB92                    INTDIV
AB92 20 A2 A2                   JSR     LDINT                   ;COMPUTE A/FR0
AB95 4C 9B AB                   JMP     SFDIV
AB98                    SPDIV
AB98 20 7B A0                   JSR     FMVPOP
AB9B                    SFDIV                                   ;X <- Y/X
AB9B 20 28 DB                   JSR     FDIV
AB9E 4C BC AB                   JMP     CRYCHK
ABA1                    INTADD
ABA1 20 A2 A2                   JSR     LDINT                   ;COMPUTE A+FR0
ABA4 4C AA AB                   JMP     SFADD
ABA7                    SPADD
ABA7 20 7B A0                   JSR     FMVPOP
ABAA                    SFADD                                   ;X <- Y+X
ABAA 20 66 DA                   JSR     FADD
ABAD 4C BC AB                   JMP     CRYCHK
ABB0                    INTSUB
ABB0 20 A2 A2                   JSR     LDINT                   ;COMPUTE A-FR0
ABB3 4C B9 AB                   JMP     SFSUB
ABB6                    SPSUB
ABB6 20 7B A0                   JSR     FMVPOP
ABB9                    SFSUB                                   ;X <- Y-X
ABB9 20 60 DA                   JSR     FSUB
ABBC                    CRYCHK
ABBC 90 09                      BCC     CRYCLR
ABBE                    CRYSND
ABBE 20 9E A2                   JSR     PCLR0                   ;CLEAR X
ABC1 A9 9B                      LDA     #CRYMSG                 ;CARRY SET => ERROR
ABC3 4C E5 9B                   JMP     ERRSUB
ABC6 38                         SEC                             ;INDICATE ERROR
ABC7                    CRYCLR
ABC7 60                         RTS
ABC8 A9 01              ONESUB  LDA     #1                      ;FR0 <- 1-FR0
ABCA D0 E4                      BNE     INTSUB                  ;JMP
ABCC 20 C8 AB           SUBONE  JSR     ONESUB                  ;FR0 <- FR0-1
ABCF 4C FE A6                   JMP     SCHGSG
```

SUBROUTINES FOR PROGRAMMABILITY

```
                        ;
ABD2                    SBSTEP                          ;BACK STEP    PC <- PC-1
ABD2 A5 C6                      LDA     PC
ABD4 38                         SEC
ABD5 E9 01                      SBC     #1
ABD7 B0 0C                      BCS     SBST10
ABD9 A6 C7                      LDX     PC+1
ABDB CA                         DEX
ABDC E0 0C                      CPX     #PRGMEM/256             ;AT BEGINNING OF PRGMEM?
ABDE B0 03                      BCS     SBST05
ABE0 4C 9D 9D                   JMP     EPERR           ;YES.  END OF PROGRAM MEM ERROR MSG AND RETURN
ABE3                    SBST05
ABE3 86 C7                      STX     PC+1
ABE5                    SBST10
ABE5 85 C6                      STA     PC
ABE7 A0 00                      LDY     #0              ;CHECK FOR NUMBER
ABE9 B1 C6                      LDA     (PC),Y
ABEB C9 96                      CMP     #NUMBER
ABED D0 20                      BNE     SBST50          ;RETURN
ABEF A5 C6                      LDA     PC              ;NUMBER => SUBTRACT MORE
ABF1 A6 C7                      LDX     PC+1
ABF3 38                         SEC
ABF4 E9 07                      SBC     #FPREC+1
ABF6 B0 05                      BCS     SBST30
ABF8 CA                         DEX
ABF9 E0 0C                      CPX     #PRGMEM/256
ABFB 90 09                      BCC     NERR
ABFD                    SBST30
ABFD 48                         PHA                     ;SAVE NEW PC
ABFE B1 C6                      LDA     (PC),Y          ;DOUBLE CHECK FOR # AT BEGINNING
AC00 A8                         TAY
AC01 68                         PLA                     ;RESTORE NEW PC
AC02 C0 96                      CPY     #NUMBER
AC04 F0 05                      BEQ     SBST40          ;OK
AC06                    NERR
AC06 A9 0D                      LDA     #KEYMSG         ;DISPLAY NOT VALID CMD ERROR MSG & RETURN
AC08 4C E5 9B                   JMP     ERRSUB
AC0B                    SBST40
AC0B 86 C7                      STX     PC+1            ;SAVE NEW PC
AC0D 85 C6                      STA     PC
AC0F                    SBST50
AC0F 60                         RTS

AC10                    SSSTEP                                  ;SINGLE STEP:  IF IN STORE PROG MODE, THEN INC PC
                        ;                                               ;IF IN IMMEDIATE MODE, EXECUTE 1 INSTRUCTION
AC10 A5 C8                      LDA     PROG
AC12 C9 01                      CMP     #STOPRG
AC14 F0 07                      BEQ     SSTP10
AC16 A9 02                      LDA     #EXEC           ;IMMEDIATE -> EXECUTE MODE
AC18 85 CB                      STA     SSTFLG
AC1A 85 C8                      STA     PROG
AC1C 60                         RTS
AC1D                    SSTP10
AC1D A0 00                      LDY     #0              ;CHECK FOR NUMBER
AC1F B1 C6                      LDA     (PC),Y
AC21 C9 96                      CMP     #NUMBER
AC23 D0 11                      BNE     SSTP20
AC25 A0 07                      LDY     #FPREC+1
AC27 B1 C6                      LDA     (PC),Y
AC29 C9 96                      CMP     #NUMBER
```

```
AC2B F0 06                  BEQ     SSTP15
AC2D 20 76 A2               JSR     PCINC
AC30 4C 06 AC               JMP     NERR
AC33                SSTP15
AC33 4C 7A A2               JMP     PCADDN
AC36                SSTP20
AC36 4C 76 A2               JMP     PCINC

AC39                SCLPRO                          ;CLEAR PROGRAM MEMORY, PC <- PRGMEM
AC39 A9 00                  LDA     #0
AC3B 85 C6                  STA     PC
AC3D A9 78                  LDA     #STP            ;OPCODE FOR STOP => INIT ALL TO STOP
AC3F A0 0C                  LDY     #PRGMEM/256
AC41 84 C7                  STY     PC+1
AC43 A2 10                  LDX     #PC1MAX+1
AC45 4C C6 A3               JMP     RAMSET

AC48                SGOTO                       ;GOTO N = 0-1023 ( 000-3FF)
AC48 20 9C A0               JSR     FPUSHO      ;SAVE X
AC4B A9 00                  LDA     #PROMSG     ;DISPLAY "ENTER PROGRAM MEM ADDRESS 0-1023"
AC4D 20 6A 9C               JSR     PUTMSG
AC50 A5 87                  LDA     DHOFLG      ;ALWAYS DECIMAL
AC52 48                     PHA
AC53 A9 00                  LDA     #0
AC55 85 87                  STA     DHOFLG
AC57 20 79 9A               JSR     LEX
AC5A A5 81                  LDA     TOKCOD
AC5C C9 96                  CMP     #NUMBER
AC5E D0 21                  BNE     SGOERR
AC60 20 D2 A2               JSR     PUTDEL
AC63 20 73 9E               JSR     FDSPO
AC66 20 D2 D9               JSR     FPI
AC69 B0 16                  BCS     SGOERR
AC6B A5 D5                  LDA     FR0+1
AC6D C9 04                  CMP     #4
AC6F B0 10                  BCS     SGOERR      ;TOO LARGE
AC71 A6 D4                  LDX     FR0
AC73 86 C6                  STX     PC
AC75 69 0C                  ADC     #PRGMEM/256
AC77 85 C7                  STA     PC+1
AC79 20 7E A0               JSR     FPOPO       ;RELOAD X
AC7C 68                     PLA
AC7D 85 87                  STA     DHOFLG      ;RESTORE DHOFLG
AC7F 18                     CLC                 ;NO ERROR
AC80 60                     RTS
AC81                SGOERR
AC81 68                     PLA
AC82 85 87                  STA     DHOFLG
AC84 20 7E A0               JSR     FPOPO       ;RELOAD X
AC87 A9 76                  LDA     #BITMSG
AC89 4C E5 9B               JMP     ERRSUB

AC8C                SLIST                           ;LIST PROGRAM STARTING WITH PC
AC8C 20 9C A0               JSR     FPUSHO      ;SAVE X
AC8F A5 CA                  LDA     DSPFLG
AC91 48                     PHA
AC92 A9 00                  LDA     #0
AC94 85 CA                  STA     DSPFLG
```

COLLEEN CALCULATOR, BY C SHAW

```
AC96                    SLSTLP
AC96 20 97 9D                   JSR     DSPRQ
AC99 B0 24                      BCS     SLST10
AC9B 20 85 A3                   JSR     PUTCRP
AC9E AD F0 02                   LDA     CRSINH
ACA1 C9 00                      CMP     #$0
ACA3 F0 13                      BEQ     BRKLST
ACA5 A5 81                      LDA     TOKCOD
ACA7 C9 96                      CMP     #NUMBER
ACA9 D0 06                      BNE     SLST05
ACAB 20 7A A2                   JSR     PCADDN
ACAE 4C B4 AC                   JMP     SLST07
ACB1                    SLST05
ACB1 20 76 A2                   JSR     PCINC
ACB4                    SLST07
ACB4 90 E0                      BCC     SLSTLP          ;OK: CONTINUE
ACB6 B0 07                      BCS     SLST10          ;END OF PROG MEM: STOP
ACB8                    BRKLST
ACB8 EE F0 02                   INC     CRSINH          ;TURN CURSOR OFF
ACBB A9 80                      LDA     #$80
ACBD 85 11                      STA     BRKKEY
ACBF                    SLST10
ACBF 68                         PLA
ACC0 85 CA                      STA     DSPFLG
ACC2 20 7E A0                   JSR     FPOPO           ;RESTORE X
ACC5 20 73 9E                   JSR     FDSPO           ;AND DISPLAY
ACC8 60                         RTS


ACC9                    SNOTRACE                        ;TRACE OFF
ACC9 A9 00                      LDA     #0
ACCB F0 02                      BEQ     STR10
ACCD                    STRACE                          ;TRACE ON
ACCD A9 01                      LDA     #1
ACCF                    STR10
ACCF 85 C9                      STA     TRACE
ACD1 A6 C8                      LDX     PROG            ;PROGRAM IN EXECUTION?
ACD3 F0 04                      BEQ     STR20           ;NO.
ACD5 49 01                      EOR     #$01            ;YES.  TRACE DETERMINES DSPFLG
ACD7 85 CA                      STA     DSPFLG
ACD9                    STR20
ACD9 60                         RTS


ACDA                    SPAUSE
ACDA A5 CA                      LDA     DSPFLG
ACDC 48                         PHA
ACDD A9 00                      LDA     #0
ACDF 85 CA                      STA     DSPFLG
ACE1 20 51 9D                   JSR     DSPSTK
ACE4 68                         PLA
ACE5 85 CA                      STA     DSPFLG
                        ;                               PAUSE FOR 30 FRAMES (1/2 SEC)
ACE7 A2 00                      LDX     #0              ;MSB
ACE9 A0 1E                      LDY     #30             ;LSB (IN FRAMES)
ACEB A9 03                      LDA     #3
ACED 8D 2A 02                   STA     CDTMF3          ;SET FLAG TO NON-ZERO
ACF0 20 5C E4                   JSR     SETVBV          ;SET TIMER
ACF3                    SPAULP
ACF3 AD 2A 02                   LDA     CDTMF3          ;WAIT FOR ZERO (TIME UP)
```

```
ACF6 D0 FB                  BNE     SPAULP
ACF8 60                     RTS
ACF9                SPROGR                          ;TO STORE PROGRAM MODE
ACF9 A9 01                  LDA     #STOPRG
ACFB C5 C8                  CMP     PROG            ;ALREADY IN MODE?
ACFD F0 09                  BEQ     SPRRTN          ;YES.
ACFF 85 C8                  STA     PROG            ;NO.
AD01 A2 5E                  LDX     #FPX            ;SAVE X REG
AD03 A0 0B                  LDY     #FPX/256
AD05 20 A7 DD               JSR     FSTOR
AD08                SPRRTN
AD08 60                     RTS

AD09                SRESET
AD09 A9 00                  LDA     #0              ;PC <- 0
AD0B 85 C6                  STA     PC
AD0D A2 0C                  LDX     #PRGMEM/256
AD0F 86 C7                  STX     PC+1
AD11 60                     RTS

AD12 20 09 AD       SRUN    JSR     SRESET          ;GOTO 0 AND RUN
AD15                SCONTI                          ;CONTINUE=> RUN STARTING AT CURRENT PC
AD15 A9 01                  LDA     #1              ;DISPLAY OFF
AD17 A6 C9                  LDX     TRACE           ;TRACE?
AD19 F0 02                  BEQ     SRUN10          ;NO.
AD1B A9 00                  LDA     #0              ;YES. DISPLAY ON
AD1D                SRUN10
AD1D 85 CA                  STA     DSPFLG
AD1F A9 02                  LDA     #EXEC
AD21 85 C8                  STA     PROG
AD23 60                     RTS

AD24                SSTP                            ;STOP PROGRAM EXECUTION
AD24                SEND                            ;END OF PROGRAM (STOP PROGRAM EXECUTION)
AD24 A9 00                  LDA     #0
AD26 85 CA                  STA     DSPFLG          ;DISPLAY ON
AD28 A5 C8                  LDA     PROG
AD2A C9 01                  CMP     #STOPRG         ;LEAVING STORE PROGRAM MODE?
AD2C D0 07                  BNE     SEND10          ;NO.
AD2E A2 5E                  LDX     #FPX            ;YES. RELOAD FRO (X)
AD30 A0 0B                  LDY     #FPX/256
AD32 20 89 DD               JSR     FLDOR
AD35                SEND10
AD35 A9 00                  LDA     #0              ;BACK TO IMMEDIATE MODE
AD37 85 C8                  STA     PROG
AD39                SNOP                    ;DO NOTHING
AD39 60                     RTS

AD3A                XLTSUB                          ;XLT R N  =>  IF X < MEM(R) THEN GOTO N
                    ;                               SUBROUTINE FOR CONDITIONAL BRANCH INSTRUCTIONS
AD3A 20 E7 A4               JSR     MEMSUB          ;WILL RETURN FROM XLTSUB IF ERROR, OTHERWISE
AD3D 20 41 A0               JSR     FLD1S           ;SAVES X ON STACK, GETS R, SETS UP X&Y.
AD40 20 B9 AB               JSR     SFSUB           ;FRO <- MEM(R) - X
AD43 A5 D4                  LDA     FRO             ;LOAD & SAVE SIGN BYTE
AD45 48                     PHA
AD46 20 7E A0               JSR     FPOPO           ;RELOAD X
AD49 68                     PLA
AD4A 18                     CLC                     ;NO ERROR
AD4B 60                     RTS
AD4C                SXEQ                            ;IF X=MEM(R) THEN GOTO N
```

COLLEEN CALCULATOR, BY C SHAW

```
AD4C 20 3A AD            JSR     XLTSUB
AD4F B0 32               BCS     XLTERR          ;ERROR
AD51 F0 2D               BEQ     MATCH
AD53 D0 1D               BNE     NOMAT
AD55                SXGE                         ;IF X>= MEM(R) THEN GOTO N
AD55 20 3A AD            JSR     XLTSUB
AD58 B0 29               BCS     XLTERR
AD5A F0 24               BEQ     MATCH
AD5C 30 22               BMI     MATCH           ;MI => MEM(R) < X => X>MEM(R)
AD5E 10 12               BPL     NOMAT
AD60                SXLT                         ;IF X<MEM(R) THEN GOTO N
AD60 20 3A AD            JSR     XLTSUB
AD63 B0 1E               BCS     XLTERR
AD65 F0 0B               BEQ     NOMAT
AD67 10 17               BPL     MATCH           ;PL => MEM(R)>=X => X<=MEM(R)
AD69 30 07               BMI     NOMAT
AD6B                SXNE                         ;IF X<>MEM(R) THEN GOTO N
AD6B 20 3A AD            JSR     XLTSUB
AD6E B0 13               BCS     XLTERR
AD70 D0 0E               BNE     MATCH
AD72                NOMAT                        ;CONDITION NOT SATISFIED
AD72 20 9C A0            JSR     FPUSHO          ;SAVE X
AD75 A9 00               LDA     #PROMSG         ;DISPLAY "ENTER PROG MEM ADR 0-1023"
AD77 20 6A 9C            JSR     PUTMSG
AD7A 20 79 9A            JSR     LEX             ;GET N AND IGNORE IT
AD7D 4C 7E A0            JMP     FPOPO           ;RESTORE X
AD80                MATCH                        ;CONDITION IS SATISFIED
AD80 4C 48 AC            JMP     SGOTO           ;GOTO N
AD83 60             XLTERR  RTS                  ;ERROR MSG ALREADY DISPLAYED BY GETMN
```

```
                        ;                       SUBROUTINE CALL & RETURN

AD84                    POPCAL
AD84 18                         CLC                     ;POP A OFF CALSTK
AD85 AE 6E 0B                   LDX     CALPTR          ;NO ERROR
AD88 D0 05                      BNE     POPC10
AD8A A9 D1                      LDA     #CLEMSG
AD8C 4C E5 9B                   JMP     ERRSUB          ;"CALL STACK EMPTY" ERROR
AD8F                    POPC10
AD8F CA                         DEX
AD90 8E 6E 0B                   STX     CALPTR
AD93 BD 80 0B                   LDA     CALSTK,X
AD96 60                         RTS

AD97                    PSHCAL
AD97 18                         CLC                     ;SAVE A ON CALSTK
AD98 AE 6E 0B                   LDX     CALPTR          ;NO ERROR
AD9B 10 05                      BPL     PSHC10          ;0-$7F IS OK
AD9D A9 DB                      LDA     #CLFMSG         ;"CALL STACK FULL" ERROR
AD9F 4C E5 9B                   JMP     ERRSUB
ADA2                    PSHC10
ADA2 9D 80 0B                   STA     CALSTK,X
ADA5 E8                         INX
ADA6 8E 6E 0B                   STX     CALPTR
ADA9 60                         RTS

ADAA                    SCALL
ADAA A6 C8                      LDX     PROG            ;CALL N (0-1023)
ADAC D0 14                      BNE     SCAL10          ;IMMEDIATE MODE?
ADAE A5 C7                      LDA     PC+1            ;NO.
ADB0 09 80                      ORA     #$80            ;YES.  PC TO BE RESTORED ON RETURN
ADB2 20 97 AD                   JSR     PSHCAL          ;SET MSBIT TO INDICATE RETURN TO IMMEDIATE MODE
ADB5 A5 C6                      LDA     PC
ADB7 20 97 AD                   JSR     PSHCAL
ADBA B0 26                      BCS     SCAL30          ;STACK FULL ERROR
ADBC 20 48 AC                   JSR     SGOTO           ;GOTO N
ADBF 4C 15 AD                   JMP     SCONTI          ;RUN SUBROUTINE (CONTINUE)
ADC2                    SCAL10
ADC2 A5 C6                      LDA     PC
ADC4 A6 C7                      LDX     PC+1            ;EXEC MODE
ADC6 18                         CLC                     ;SKIP PAST N
ADC7 69 08                      ADC     #FPREC+2        ;TO GET RETURN ADDRESS.
ADC9 90 01                      BCC     SCAL20
ADCB E8                         INX
ADCC                    SCAL20
ADCC 48                         PHA
ADCD 8A                         TXA
ADCE 20 97 AD                   JSR     PSHCAL          ;PC+1
ADD1 68                         PLA
ADD2 20 97 AD                   JSR     PSHCAL          ;PC
ADD5 B0 0B                      BCS     SCAL30          ;STACK FULL ERROR => DON'T GO
ADD7 20 48 AC                   JSR     SGOTO
ADDA 90 06                      BCC     SCAL30
ADDC 20 84 AD                   JSR     POPCAL          ;ERROR => THROW AWAY RETURN ADDR FROM STACK
ADDF 20 84 AD                   JSR     POPCAL          ;& KEEP OLD PC
ADE2 60                 SCAL30  RTS


ADE3                    SRETUR                          ;RETURN => POP PC OFF STACK, GOTO PC
ADE3 20 84 AD                   JSR     POPCAL          ;PC
ADE6 B0 12                      BCS     SRET20          ;ERROR - STACK EMPTY
```

COLLEEN CALCULATOR, BY C SHAW

```
ADE8 85 C6                  STA     PC
ADEA 20 84 AD               JSR     POPCAL          ;PC+1
ADED B0 0B                  BCS     SRET20          ;STACK EMPTY => DON'T EXECUTE RETURN
ADEF 10 07                  BPL     SRET10
ADF1 29 7F                  AND     #$7F            ;PC+1 (MSB) <0  =>  RETURN TO IMMEDIATE MODE
ADF3 85 C7                  STA     PC+1
ADF5 4C 24 AD               JMP     SSTP
ADF8                SRET10
ADF8 85 C7                  STA     PC+1            ;PC+1 >0  => STAY IN EXEC MODE
ADFA                SRET20
ADFA 60                     RTS
```

INSERT & DELETE

```
                        ;
ADFB                    SDELET          ;DELETE - FOR I=PC TO 1022+PRGMEM:MEM(I)<-MEM(I+1):NEXT I
                        ;                       MEM(1023+PRGMEM)<-STP
ADFB A5 C6                      LDA     PC                      ;MOVE PC TO TEMP PTR
ADFD 85 95                      STA     JMPTR1
ADFF A5 C7                      LDA     PC+1
AE01 85 96                      STA     JMPTR1+1
AE03 A0 01              SDELP1  LDY     #1
AE05 B1 95                      LDA     (JMPTR1),Y      ;MEM(I+1)
AE07 88                         DEY
AE08 91 95                      STA     (JMPTR1),Y      ;MEM(I)
AE0A E6 95                      INC     JMPTR1
AE0C D0 F5                      BNE     SDELP1          ;CONTINUE
AE0E E6 96                      INC     JMPTR1+1
AE10 A5 96                      LDA     JMPTR1+1
AE12 C9 10                      CMP     #PC1MAX+1       ;AT END OF MEM?
AE14 D0 ED                      BNE     SDELP1          ;NO.  CONTINUE

AE16 A9 78                      LDA     #STP            ;DONE   STORE "STOP"
AE18 8D FF 0F                   STA     PRGMEM+PRGLEN-1
AE1B 60                         RTS

AE1C                    SINSER          ;INSERT - FOR I=1022+PRGMEM TO PC:MEM(I+1)<-MEM(I):NEXT I
                        ;               MEM(PC)<-STP
AE1C A9 FE                      LDA     #$FE            JMPTR1<-ADDR(END OF PRGMEM-1)
AE1E 85 95                      STA     JMPTR1
AE20 A9 0F                      LDA     #PC1MAX
AE22 85 96                      STA     JMPTR1+1
AE24                    SINSLP
AE24 A0 00                      LDY     #0
AE26 B1 95                      LDA     (JMPTR1),Y      ;MEM(I)
AE28 C8                         INY
AE29 91 95                      STA     (JMPTR1),Y      ;MEM(I+1)
AE2B C6 95                      DEC     JMPTR1
AE2D A6 95                      LDX     JMPTR1
AE2F E0 FF                      CPX     #$FF
AE31 D0 02                      BNE     INS10
AE33 C6 96                      DEC     JMPTR1+1
AE35                    INS10
AE35 A4 96                      LDY     JMPTR1+1
AE37 C4 C7                      CPY     PC+1
AE39 90 06                      BCC     INS30           ;JMPTR1<PC  =>  STOP
AE3B D0 E7                      BNE     SINSLP          ;JMPTR1>PC  => CONTINUE
AE3D E4 C6                      CPX     PC              ;JMPTR1+1 = PC+1
AE3F B0 E3                      BCS     SINSLP          ;JMPTR1 >= PC  => CONTINUE
AE41                    INS30
AE41 A0 00                      LDY     #0              ;DONE MEM(PC) <- STP
AE43 A9 78                      LDA     #STP
AE45 91 C6                      STA     (PC),Y
AE47 60                         RTS
```

```
                        ;                       OPEN & CLOSE

AE48                    FCLOSE                                  ;CLOSE TIOCB
AE48 A2 30                      LDX     #TIOCB
AE4A A9 0C                      LDA     #CLOSE
AE4C 9D 42 03                   STA     ICCOM,X
AE4F D0 5A                      BNE     CIOCAL          ;JMP TO CIO CALL

AE51 A9 04              OPNIN   LDA     #INPUT          ;OPEN FILE FOR INPUT
AE53 D0 02                      BNE     FOPEN           ;JMP
AE55 A9 08              OPNOUT  LDA     #OUTPUT         ;OPEN FINE FOR OUTPUT
AE57 48                 FOPEN   PHA                     ;OPEN FILE FOR INPUT/OUTPUT (ACCORDING TO A)
AE58 A5 C8                      LDA     PROG
AE5A F0 05                      BEQ     FOP10
AE5C A9 0D                      LDA     #KEYMSG         ;EXEC MODE => NOT VALID COMMAND
AE5E 4C E5 9B                   JMP     ERRSUB
AE61                    FOP10
AE61 A9 51                      LDA     #FSPMSG         ;"ENTER FILE SPEC (NO QUOTES)" MESSAGE
AE63 A0 B9                      LDY     #FSPMSG/256
AE65 20 6C 9C                   JSR     PTMSG2
AE68                    FOPLP1
AE68 20 62 9A                   JSR     LXINIT          ;SET UP CURSOR, DISPLAY '>'
AE6B 20 E8 A0           FOPLP3  JSR     GTCHR           ;REMOVE LEADING SEPARATORS
AE6E C9 20                      CMP     #'
AE70 F0 F9                      BEQ     FOPLP3
AE72 E6 82                      INC     TOKPTR          ;SAVE FIRST CHAR
AE74                    FOPLP2
AE74 20 E8 A0                   JSR     GTCHR
AE77 C9 20                      CMP     #'
AE79 F0 14                      BEQ     FOP30           ;DONE
AE7B C9 9C                      CMP     #DELLIN
AE7D F0 E9                      BEQ     FOPLP1          ;TRY AGAIN
AE7F A6 82                      LDX     TOKPTR
AE81 E0 10                      CPX     #NUMLEN
AE83 90 06                      BCC     FOP20
AE85 68                         PLA                     ;POP INPUT/OUTPUT INDICATOR
AE86 A9 82                      LDA     #DIGMSG         ;"TOO MANY CHARS" ERROR MESSAGE
AE88 4C E5 9B                   JMP     ERRSUB
AE8B                    FOP20
AE8B E6 82                      INC     TOKPTR          ;SAVE CHAR
AE8D D0 E5                      BNE     FOPLP2          ;JMP (CONTINUE)
AE8F                    FOP30                           ;NOW HAVE FILESPEC STRING IN TOKBUF
AE8F A9 9B                      LDA     #CR
AE91 A6 82                      LDX     TOKPTR
AE93 9D 00 05                   STA     TOKBUF,X        ;PUT CR AT END OF STRING
AE96 A2 30                      LDX     #TIOCB          ;TEMPORARY IOCB #
AE98 A9 03                      LDA     #OPEN
AE9A 9D 42 03                   STA     ICCOM,X
AE9D A9 00                      LDA     #TOKBUF
AE9F 9D 44 03                   STA     ICBAL,X
AEA2 A9 05                      LDA     #TOKBUF/256
AEA4 9D 45 03                   STA     ICBAH,X
AEA7 68                         PLA                     ;INPUT OR OUTPUT
AEA8 9D 4A 03                   STA     ICAX1,X
AEAB                    CIOCAL                          ;CALL CIO AND CHECK FOR ERROR
AEAB 20 56 E4                   JSR     CIOV
AEAE C0 01                      CPY     #SUCCES
AEB0 D0 02                      BNE     IOERR           ;ERROR
AEB2 18                         CLC                     ;NO ERROR
AEB3 60                         RTS
```

```
AEB4                    IOERR                           ;DISPLAY "ERROR - "  <ERROR #>
AEB4 98                         TYA                     ;Y=ERROR #
AEB5 48                         PHA                     ;SAVE
AEB6 20 13 9C                   JSR     ERRSB2          ;DISPLAY "ERROR - ", DO OTHER STUFF
AEB9 20 93 A3                   JSR     PTCHSP
AEBC 20 9C A0                   JSR     FPUSHO          ;SAVE X
AEBF 68                         PLA                     ;CONVERT ERROR # TO FP TO ASCII (0-255)
AEC0 85 D4                      STA     FR0
AEC2 A9 00                      LDA     #0
AEC4 85 D5                      STA     FR0+1
AEC6 20 AA D9                   JSR     IFP
AEC9 20 E6 D8                   JSR     FASC
AECC A0 FF                      LDY     #$FF            ;FIND END OF TEXT
AECE C8                 IOELP   INY
AECF B1 F3                      LDA     (INBUFF),Y
AED1 10 FB                      BPL     IOELP
AED3 29 7F                      AND     #$7F            ;REMOVE END OF TEXT INDICATOR
AED5 91 F3                      STA     (INBUFF),Y
AED7 C8                         INY
AED8 98                         TYA
AED9 AA                         TAX                     ;NUMBER OF CHARS
AEDA A5 F3                      LDA     INBUFF
AEDC A4 F4                      LDY     INBUFF+1
AEDE 20 BA A3                   JSR     PTTXTP          ;DISPLAY & PRINT WITH <RETURN>
AEE1 20 7E A0                   JSR     FPOPO           ;RELOAD X
AEE4 38                         SEC                     ;ERROR
AEE5 60                 MSAV20  RTS


AEE6                    SPLOAD                          ;LOAD PROGRAM MEM FROM SPECIFIED FILE
AEE6 20 51 AE                   JSR     OPNIN
AEE9 B0 FA                      BCS     MSAV20          ;ERROR IF CRY SET
AEEB A9 07                      LDA     #GETCHR
AEED D0 07                      BNE     PSAV10          ;JMP

AEEF                    SPSAVE                          ;SAVE PROGRAM MEM IN FILE
AEEF 20 55 AE                   JSR     OPNOUT
AEF2 B0 F1                      BCS     MSAV20
AEF4 A9 0B                      LDA     #PUTCHR
AEF6                    PSAV10
AEF6 A2 30                      LDX     #TIOCB
AEF8 9D 42 03                   STA     ICCOM,X
AEFB A9 00                      LDA     #PRGMEM
AEFD 9D 44 03                   STA     ICBAL,X
AF00 A9 0C                      LDA     #PRGMEM/256
AF02 9D 45 03                   STA     ICBAH,X
AF05 A9 00                      LDA     #PRGLEN
AF07 9D 48 03                   STA     ICBLL,X
AF0A A9 04                      LDA     #PRGLEN/256
AF0C 9D 49 03                   STA     ICBLH,X
AF0F D0 29                      BNE     MSAV15          ;(JMP) CALL CIO & CLOSE

AF11                    SMLOAD                          ;LOAD MEM FROM FILE
AF11 20 51 AE                   JSR     OPNIN           ;OPEN FILE FOR INPUT
AF14 B0 CF                      BCS     MSAV20
AF16 A9 07                      LDA     #GETCHR
AF18 D0 07                      BNE     MSAV10          ;JMP
AF1A                    SMSAVE                          ;SAVE ALL OF MEM IN SPECIFIED FILE
```

```
AF1A 20 55 AE              JSR     OPNOUT          ;OPEN FILE FOR OUTPUT
AF1D B0 C6                 BCS     MSAV20          ;ERROR - RETURN
AF1F A9 0B                 LDA     #PUTCHR
AF21 A2 30        MSAV10   LDX     #TIOCB
AF23 9D 42 03              STA     ICCOM,X
AF26 A9 00                 LDA     #MEMORY
AF28 9D 44 03              STA     ICBAL,X
AF2B A9 08                 LDA     #MEMORY/256
AF2D 9D 45 03              STA     ICBAH,X
AF30 A9 58                 LDA     #MEMLEN*FPREC
AF32 9D 48 03              STA     ICBLL,X
AF35 A9 02                 LDA     #MEMLEN*FPREC/256
AF37 9D 49 03              STA     ICBLH,X
AF3A 20 AB AE     MSAV15   JSR     CIOCAL          ;CALL CIO AND CHECK FOR ERROR
AF3D 4C 48 AE              JMP     FCLOSE          ;CLOSE FILE
```

```
                        ;                    SUBROUTINES FOR CONVERSIONS

AF40                    SMETER
AF40 A2 42                      LDX     #0*FPREC+LENGTH ;LOAD LSB OF ADDR OF CONVERSION CONSTANT
AF42 D0 1A                      BNE     LENG            ;JMP
AF44                    SINCHE
AF44 A2 48                      LDX     #1*FPREC+LENGTH
AF46 D0 16                      BNE     LENG
AF48                    SFEET
AF48 A2 4E                      LDX     #2*FPREC+LENGTH
AF4A D0 12                      BNE     LENG
AF4C A2 54              SYARDS  LDX     #3*FPREC+LENGTH
AF4E D0 0E                      BNE     LENG
AF50 A2 5A              SMILES  LDX     #4*FPREC+LENGTH
AF52 D0 0A                      BNE     LENG
AF54 A2 60              SCM     LDX     #5*FPREC+LENGTH
AF56 D0 06                      BNE     LENG
AF58 A2 66              SKM     LDX     #6*FPREC+LENGTH
AF5A D0 02                      BNE     LENG
AF5C A2 6C              SNAUTI  LDX     #7*FPREC+LENGTH
AF5E                    LENG
AF5E A9 E5                      LDA     #ZM             ;LSB OF MESSAGE ADDR
AF60 D0 2C                      BNE     CONVRT          ;JMP

AF62                    SKG
AF62 A2 42                      LDX     #LENGTH         ;CONSTANT 1 (NO CONVERSION)
AF64 D0 0A                      BNE     MAS             ;JMP
AF66 A2 72              SOZ     LDX     #0*FPREC+MASS
AF68 D0 06                      BNE     MAS
AF6A                    SLB
AF6A A2 78              SPOUND  LDX     #1*FPREC+MASS
AF6C D0 02                      BNE     MAS
AF6E                    SGM
AF6E A2 7E              SGRAMS  LDX     #2*FPREC+MASS
AF70                    MAS
AF70 A9 E9                      LDA     #ZKG
AF72 D0 1A                      BNE     CONVRT          ;JMP

AF74 A2 42              SFLOZ   LDX     #LENGTH
AF76 D0 06                      BNE     VOL
                                .IF     ASMBL
                        STSP    LDX     #0*FPREC+VOLUME
                                BNE     VOL
                        STBSP   LDX     #1*FPREC+VOLUME
                                BNE     VOL
                        SCUPS   LDX     #2*FPREC+VOLUME
                                BNE     VOL
                        SQUART  LDX     #3*FPREC+VOLUME
                                BNE     VOL
                                .ENDIF
AF78 A2 84              SGAL    LDX     #0*FPREC+VOLUME
AF7A D0 02                      BNE     VOL
AF7C A2 8A              SLITER  LDX     #1*FPREC+VOLUME
AF7E                    VOL
AF7E A9 F0                      LDA     #ZFL
AF80 D0 0C                      BNE     CONVRT          ;JMP
AF82 A2 42              SCDEG   LDX     #LENGTH         ;1
AF84 D0 06                      BNE     CDGR            ;JMP
AF86 A2 90              SCGRAD  LDX     #0*FPREC+DEGREE
AF88 D0 02                      BNE     CDGR
AF8A A2 96              SCRAD   LDX     #1*FPREC+DEGREE
```

```
AF8C A9 F8          CDGR    LDA     #ZDEG           ;DEGREES MSG

AF8E                CONVRT                          ;CONVERT TO DIFFERENT UNITS
AF8E A4 D2                  LDY     CONFLG          ;FLAG SET?
AF90 D0 1D                  BNE     CONV10          ;YES.

AF92 85 D2                  STA     CONFLG          ;NO. SAVE MSG ADDR LSB & DO INTERMEDIATE CONVERSION
AF94 A0 B6                  LDY     #LENGTH/256     ;LOAD CONVERSION CONSTANT
AF96 20 98 DD               JSR     FLD1R
AF99 20 8C AB               JSR     SFMUL           ;AND MULTIPLY.
AF9C B0 24                  BCS     CONVRN          ;OVERFLOW ERROR
AF9E A9 6F                  LDA     #CN1MSG         ;DISPLAY "INTERMEDIATE UNITS" MESSAGE
AFA0 20 6A 9C               JSR     PUTMSG
AFA3 A5 D2                  LDA     CONFLG          ;DISPLAY NEW UNITS
AFA5 A0 B8                  LDY     #ZM/256         ;LOAD PAGE #
AFA7 20 6C 9C               JSR     PTMSG2
AFAA A9 7B                  LDA     #CN2MSG         ;DISPLAY "ENTER NEW UNITS"
AFAC 4C 6A 9C               JMP     PUTMSG

AFAF                CONV10                          ;CONVERT FROM INTERMEDIATE UNITS TO FINAL UNITS
AFAF C5 D2                  CMP     CONFLG          ;FLAGS MATCH?
AFB1 D0 10                  BNE     CONERR          ;NO. ERROR - CAN'T MIX TYPES
AFB3 A0 B6                  LDY     #LENGTH/256     ;YES. LOAD CONVERSION CONSTANT
AFB5 20 98 DD               JSR     FLD1R
AFB8 20 9B AB               JSR     SFDIV           ;AND DIVIDE.
AFBB B0 05                  BCS     CONVRN          ;OVERFLOW
AFBD A9 86                  LDA     #CN3MSG         ;DISPLAY "CONVERSION COMPLETE"
AFBF 20 6A 9C               JSR     PUTMSG
AFC2                CONVRN
AFC2 60                     RTS
AFC3                CONERR
AFC3 A9 C9                  LDA     #UNIMSG
AFC5 4C E5 9B               JMP     ERRSUB          ;DISPLAY "UNIT MISMATCH" ERROR MESSAGE & RETURN
```

```
                        ;                          COMPOUND INTEREST SUBROUTINES

      AFC8              LDBAL                                    ;FRO<-BAL
      AFC8 A9 04                  LDA     #4
      AFCA D0 12                  BNE     MEMLOD                 ;JMP
      AFCC A9 05        LDFV      LDA     #5                     ;FRO<-FV
      AFCE D0 0E                  BNE     MEMLOD
      AFD0 A9 06        LDI       LDA     #6                     ;FRO<-I
      AFD2 D0 0A                  BNE     MEMLOD
      AFD4 A9 07        LDN       LDA     #7                     ;FRO<-N
      AFD6 D0 06                  BNE     MEMLOD
      AFD8 A9 08        LDPMT     LDA     #8                     ;FRO<-PMT
      AFDA D0 02                  BNE     MEMLOD
      AFDC A9 09        LDPV      LDA     #9                     ;FRO<-PV
      AFDE              MEMLOD                                   ;FRO<-MEM(A)
      AFDE 85 AF                  STA     MEMNUM
      AFE0 20 D7 A4               JSR     MEMLDR
      AFE3 4C 89 DD               JMP     FLDOR
```

```
                        ;                       SUBROUTINES TO COMPUTE PARTS OF COMPOUND INT. EQUATIONS
AFE6 20 D0 AF     Z1PLI JSR     LDI             ;COMPUTE (1+I)
AFE9 A9 01            LDA     #1
AFEB 4C A1 AB         JMP     INTADD
AFEE 20 E6 AF     Z1IN1 JSR     Z1PLI           ;COMPUTE (1+I)^(N-1)
AFF1 20 9C A0         JSR     FPUSHO          ;SAVE (1+I)
AFF4 20 D4 AF         JSR     LDN
AFF7 20 CC AB         JSR     SUBONE          ;N-1
AFFA 4C 76 A9         JMP     SPOWER
AFFD 20 E6 AF     Z1IN  JSR     Z1PLI           ;COMPUTE (1+I)^N
B000 20 9C A0         JSR     FPUSHO
B003 20 D4 AF         JSR     LDN
B006 4C 76 A9         JMP     SPOWER
B009 20 FD AF     Z1IMN JSR     Z1IN            ;(1+I)^-N = 1/((1+I)^N)
B00C 4C 03 AA         JMP     SRECIP
B00F 20 FD AF     Z1INM1 JSR    Z1IN            ;((1+I)^N)-1
B012 4C CC AB         JMP     SUBONE
B015 20 0F B0     Z1IN1I JSR    Z1INM1          ;((1+I)^N-1)/I
B018 20 9C A0     DIVI  JSR     FPUSHO          ;FR0 <- FR0/I
B01B 20 D0 AF         JSR     LDI
B01E 4C 98 AB         JMP     SPDIV
B021 20 E6 AF     ZLN1I JSR     Z1PLI           ;LN(1+I)
B024 4C 7C AB         JMP     SLN
B027 20 09 B0     Z11IMN JSR    Z1IMN           ;1-(1+I)^-N
B02A 4C C8 AB         JMP     ONESUB
B02D 20 27 B0     Z11INI JSR    Z11IMN          ;(1-(1+I)^-N)/I
B030 4C 18 B0         JMP     DIVI
B033 AD 67 0B     ZMUL1I LDA    DUEFLG          ;IF ANNUITY DUE THEN FR0 <- FR0 * (1+I)
B036 29 01            AND     #1
B038 D0 09            BNE     ZMRTN
B03A 20 9C A0         JSR     FPUSHO
B03D 20 E6 AF         JSR     Z1PLI
B040 4C 89 AB         JMP     SPMUL
B043 60           ZMRTN RTS
```

COLLEEN CALCULATOR, BY C SHAW

```
B044 A9 00        SANNUI  LDA     #0          ;ANNUITY
B046 F0 02                BEQ     SCMP10      ;JMP
B048 A9 01        SCMPND  LDA     #1          ;COMPOUND INTEREST
B04A 8D 68 0B     SCMP10  STA     ANNFLG
B04D 60                   RTS
B04E A9 00        SFVDUE  LDA     #0          ;ANNUITY DUE (PAY AT BEGINNING OF PERIOD
B050 F0 02                BEQ     SORD10      ;E.G. SAVINGS ACCT.)
B052 A9 01        SFVORD  LDA     #1          ;ORDINARY ANNUITY (PAY AT END OF PERIOD
B054 8D 67 0B     SORD10  STA     DUEFLG      ;E.G. LOAN
B057 60                   RTS
B058 A9 02        SPVDUE  LDA     #2          ;ANNUITY DUE/PV
B05A D0 F8                BNE     SORD10
B05C A9 03        SPVORD  LDA     #3          ;ORDINARY ANNUITY/PV
B05E D0 F4                BNE     SORD10
B060 A9 00        SENTER  LDA     #0          ;ENTER VALUE
B062 F0 02                BEQ     SFND10
B064 A9 01        SFIND   LDA     #1          ;FIND VALUE, GIVEN OTHER VARIABLES
B066 8D 69 0B     SFND10  STA     ENTFLG
B069 60                   RTS
```

```
B06A                    SBAL                                ;BAL = BALLOON PAYMENT
B06A AD 69 0B                   LDA     ENTFLG
B06D D0 0C                      BNE     SBAL10
B06F                    SBAL05                              ;ENTER VALUE
B06F A9 04                      LDA     #4
B071                    MEMSTO
B071 85 AF                      STA     MEMNUM
B073 A9 02                      LDA     #2                  ;FIX 2 (DISPLAYING DOLLAR VALUE)
B075 20 23 A8                   JSR     SFIX2
B078 4C B6 AA                   JMP     SSTO10              ;STORE REG & DISPLAY
B07B                    SBAL10                              ;FIND VALUE
B07B AD 68 0B                   LDA     ANNFLG
B07E F0 05                      BEQ     SBAL20
B080                    SBAL15
B080 A9 0D                      LDA     #KEYMSG             ;NO BAL PAYMENT IF COMPOUNT INTEREST
B082 4C E5 9B                   JMP     ERRSUB
B085                    SBAL20
B085 AD 67 0B                   LDA     DUEFLG
B088 29 02                      AND     #2
B08A F0 F4                      BEQ     SBAL15              ;NO BAL IF FV
                        ;                                   BAL = (PV - PMT*(1-(1+I)^-N)/I)/(1+I)^-N
                        ;               ANNUITY DUE:        BAL = (PV - PMT*(1+I)*(1-(1+I)^-N)/I)/(1+I)^-N
B08C 20 D8 AF                   JSR     LDPMT               ;FRO <- PMT
B08F 20 33 B0                   JSR     ZMUL1I              ;IF ANNUITY DUE THEN FRO <- FRO * (1+I)
B092 20 9C A0                   JSR     FPUSHO
B095 20 2D B0                   JSR     Z11INI              ;(1-(1+I)^-N)/I
B098 20 89 AB                   JSR     SPMUL
B09B 20 B6 DD                   JSR     FMOVE
B09E 20 DC AF                   JSR     LDPV                ;PV
B0A1 20 B9 AB                   JSR     SFSUB
B0A4 20 9C A0                   JSR     FPUSHO
B0A7 20 09 B0                   JSR     Z1IMN               ;81+I)^-N
B0AA 20 98 AB                   JSR     SPDIV
B0AD 4C 6F B0                   JMP     SBAL05              ;STORE NEW BAL

B0B0                    SFV                                 ;FV = FUTURE VALUE
B0B0 AD 69 0B                   LDA     ENTFLG
B0B3 D0 05                      BNE     SFV10
B0B5 A9 05              SFV05   LDA     #5
B0B7 4C 71 B0                   JMP     MEMSTO              ;ENTER
B0BA                    SFV10                               ;FIND
B0BA AD 68 0B                   LDA     ANNFLG
B0BD F0 0B                      BEQ     SFV20
                        ;                                   COMPOUND INTEREST FV=PV*(1+I)^N
B0BF 20 FD AF                   JSR     Z1IN                ;(1+I)^N
B0C2 A9 09                      LDA     #9                  ;PV
B0C4 20 FE A4                   JSR     MEMMUL              ;FRO <- FRO * MEM(A)
B0C7 4C B5 B0                   JMP     SFV05               ;STORE NEW FV
B0CA                    SFV20                               ;ORDINARY ANNUITY FV=PMT*((1+I)^N-1)/I
                        ;                                   ;ANNUITY DUE      FV=ABOVE * (1+I)
B0CA 20 15 B0                   JSR     Z1IN1I              ;((1+I)^N-1)/I
B0CD A9 08                      LDA     #8                  ;PMT
B0CF 20 FE A4                   JSR     MEMMUL
B0D2 20 33 B0                   JSR     ZMUL1I              ;IF ANNUITY DUE THEN FRO <- FRO * (1+I)
B0D5 4C B5 B0                   JMP     SFV05               ;STORE NEW FV

B0D8                    SI                                  ;ENTERED I = INTEREST IN PERCENT
B0D8 AD 69 0B                   LDA     ENTFLG
B0DB D0 21                      BNE     SI10
```

```
BODD 20 9C A0              JSR     FPUSHO          ;CONVERT INTEREST IN PERCENT TO FRACTIONAL VALUE
BOE0 A9 64                 LDA     #100            ;BY DIVIDING BY 100.
BOE2 20 A7 A2              JSR     PSETO
BOE5 20 98 AB              JSR     SPDIV
BOE8 A9 09          SI05   LDA     #9              ;FIX 9 FOR I/100
BOEA 20 23 A8              JSR     SFIX2
BOED A9 06                 LDA     #6              ;I
BOEF 85 AF                 STA     MEMNUM
BOF1 20 B6 AA              JSR     SSTO10
BOF4 A9 64                 LDA     #100
BOF6 20 83 AB              JSR     INTMUL
BOF9 A9 04                 LDA     #4              ;FIX 4 FOR I IN PERCENT (DISPLAY)
BOFB 4C 23 A8              JMP     SFIX2           ;AND RETURN
BOFE                SI10
BOFE AD 68 0B              LDA     ANNFLG
B101 F0 1E                 BEQ     SI20
                    ;                              COMPOUND INTEREST I=((FV/PV)^(1/N)-1)*100
B103 20 DC AF              JSR     LDPV
B106 20 B6 DD              JSR     FMOVE
B109 20 CC AF              JSR     LDFV
B10C 20 9B AB              JSR     SFDIV           ;FV/PV
B10F 20 9C A0              JSR     FPUSHO
B112 20 D4 AF              JSR     LDN
B115 20 03 AA              JSR     SRECIP          ;1/N
B118 20 76 A9              JSR     SPOWER
B11B 20 CC AB              JSR     SUBONE
B11E 4C E8 B0              JMP     SI05            ;STORE NEW I
B121                SI20                           ;ANNUITY - USE NEWTON - RAPHSON ITERATION (SEE SSQRT)
                           .IF     ASMBL
                           LDA     DUEFLG
                           CMP     #1
                           BEQ     SI30
                           .ENDIF
B121 A9 0D                 LDA     #KEYMSG         ;NOT VALID COMMAND - NOT IMPLEMENTED YET
B123 4C E5 9B              JMP     ERRSUB
                           .IF     ASMBL
                    SI30                           ;ORDINARY ANNUITY/FV
                    ;      F(I) = PMT*((1+I)^N-1)/I - FV = 0
                    ;      FPRIME(I) = (PMT*N*(1+I)^(N-1)-(F(I)+FV))/I
                    ;      DELTA I = F(I)/FPRIME(I)
                           LDA     #0              ;I = MEM(6) <- .01 = $3F,1,0,0,0,0
                           LDX     #3
                    SILP1
                           STA     6*FPREC+MEMORY+2,X
                           DEX
                           BPL     SILP1
                           LDA     #$3F
                           STA     6*FPREC+MEMORY
                           LDA     #1
                           STA     6*FPREC+MEMORY+1
                           STA     DMFLG           ;DON'T DISPLAY MEM DURING ITERATION (<-1)
                           LDA     #$FF            ;NUMBER OF ITERATIONS
                           STA     ITER
                    SILP
                           JSR     Z1IN1I          ;((1+I)^N-1)/I
                           LDA     #8              ;PMT
                           JSR     MEMMUL
                           JSR     FPUSHO          ;SAVE FOR SPSUB
                           JSR     FSTOT           ;SAVE FOR F'(I)
                           JSR     LDFV
```

```
                        JSR     SPSUB
                        JSR     FPUSH0          ;SAVE F(I)
                        JSR     Z1IN1           ;(1+I)^(N-1)
                        LDA     #8              ;PMT
                        JSR     MEMMUL
                        LDA     #7              ;N
                        JSR     MEMMUL
                        JSR     FLD1T           ;RELOAD F(I)-PV
                        JSR     SFSUB
                        JSR     FPUSH0
                        JSR     LDI
                        JSR     SPDIV           ;F'(I)
                        JSR     SPDIV           ;F(I)/F'(I) = DELTA I
                        LDA     FR0             ;0?
                        BNE     SI35            ;NO.  CONTINUE
                        JSR     LDI             ;YES.  RELOAD I INTO X REG
                        JMP     SI40            ;DONE
                SI35
                        JSR     FMOVE           ;NO.  I<- I+DELTA I
                        JSR     LDI
                        JSR     SFSUB
                        DEC     ITER
                        BEQ     SI40            ;DONE
                        JSR     SI05            ;STORE NEW I
                        JMP     SILP            ;CONTINUE
                SI40    DEC     DMFLG           ; <- 0   DONE => RETURN
                        JMP     SI05            ;STORE NEW I & DISPLAY
                        .ENDIF

B126            SN                              ;N = NUMBER OF PERIODS
B126 AD 69 0B           LDA     ENTFLG
B129 D0 05              BNE     SN10
B12B A9 07      SN05    LDA     #7
B12D 4C 71 B0           JMP     MEMSTO          ;ENTER
B130            SN10                            ;FIND
B130 AD 68 0B           LDA     ANNFLG
B133 F0 1B              BEQ     SN20
B135 20 DC AF           JSR     LDPV            ;COMPOUND INTEREST N = LN(FV/PV)/LN(1+I)
B138 20 B6 DD           JSR     FMOVE
B13B 20 CC AF           JSR     LDFV
B13E 20 9B AB           JSR     SFDIV           ;FV/PV
B141 20 7C A8           JSR     SLN
B144 20 9C A0           JSR     FPUSH0
B147 20 21 B0           JSR     ZLN1I           ;LN(1+I)
B14A 20 98 AB           JSR     SPDIV
B14D 4C 2B B1           JMP     SN05            ;STORE NEW N
B150            SN20                            ;ANNUITY FV OR PV?
B150 AD 67 0B           LDA     DUEFLG
B153 29 02              AND     #2
B155 D0 28              BNE     SN50            ;1=> PV
                ;                                0=> FV  N=LN(FV*I/PMT+1)/LN(1+I)
                ;                               DUE      N=LN(FV*I/(PMT*(1+I))+1)/LN(1+I)
B157 20 CC AF           JSR     LDFV
B15A A9 06              LDA     #6              ;I
B15C 20 FE A4           JSR     MEMMUL
B15F 20 9C A0           JSR     FPUSH0
B162 20 D8 AF           JSR     LDPMT
B165 20 33 B0           JSR     ZMUL1I          ;IF DUE THEN *(1+I)
B168 20 98 AB           JSR     SPDIV
B16B A9 01              LDA     #1
```

```
B16D 20 A1 AB             JSR     INTADD
B170 20 7C A8             JSR     SLN
B173 20 9C A0             JSR     FPUSHO
B176 20 21 B0             JSR     ZLN1I           ;LN(1+I)
B179 20 98 AB             JSR     SPDIV
B17C 4C 2B B1             JMP     SNO5
B17F              SN50
                  ;                                PV N = LN((PMT-I*BAL)/(PMT-I*PV))/LN(1+I)
                  ;                               DUE N = LN((PMT*(1+I)-I*BAL)/(PMT*(1+I)-I*PV))/LN(1+I)
B17F 20 D8 AF             JSR     LDPMT
B182 20 33 B0             JSR     ZMUL1I          ;IF DUE THEN *(1+I)
B185 20 55 A0             JSR     FSTOT
B188 20 9C A0             JSR     FPUSHO
B18B 20 D0 AF             JSR     LDI
B18E 20 B6 DD             JSR     FMOVE
B191 A9 04                LDA     #4              ;BAL
B193 20 FE A4             JSR     MEMMUL
B196 20 B6 AB             JSR     SPSUB
B199 20 9C A0             JSR     FPUSHO
B19C 20 D0 AF             JSR     LDI
B19F A9 09                LDA     #9              ;PV
B1A1 20 FE A4             JSR     MEMMUL
B1A4 20 B6 DD             JSR     FMOVE
B1A7 20 47 A0             JSR     FLDOT           ;RELOAD PMT OR PMT*(1+I)
B1AA 20 B9 AB             JSR     SFSUB
B1AD 20 98 AB             JSR     SPDIV
B1B0 20 7C A8             JSR     SLN
B1B3 20 9C A0             JSR     FPUSHO
B1B6 20 21 B0             JSR     ZLN1I           ;LN(1+I)
B1B9 20 98 AB             JSR     SPDIV
B1BC 4C 2B B1             JMP     SNO5

B1BF              SPMT                            ;PMT = PAYMENT
B1BF AD 69 0B             LDA     ENTFLG
B1C2 D0 05                BNE     SPMT10
B1C4 A9 08        SPMT05  LDA     #8
B1C6 4C 71 B0             JMP     MEMSTO          ;ENTER
B1C9              SPMT10                          ;FIND
B1C9 AD 68 0B             LDA     ANNFLG
B1CC F0 03                BEQ     SPMT20
B1CE 4C 80 B0             JMP     SBAL15          ;NO PMT IF COMPOUNT INTEREST - NOT VALID COMMAND
B1D1              SPMT20
B1D1 AD 67 0B             LDA     DUEFLG
B1D4 29 02                AND     #2
B1D6 D0 17                BNE     SPMT30
                  ;                                FV PMT = FV*I/((1+I)^N-1)
                  ;                               DUE PMT = FV*I/((1+I)^N-1)*(1+I)=ABOVE/(1+I)
B1D8 20 0F B0             JSR     Z1INM1          ;(1+I)^N-1
B1DB 20 33 B0             JSR     ZMUL1I          ;IF DUE THEN *(1+I)
B1DE 20 B6 DD             JSR     FMOVE
B1E1 20 CC AF             JSR     LDFV
B1E4 20 9B AB             JSR     SFDIV
B1E7 A9 06                LDA     #6              ;I
B1E9 20 FE A4             JSR     MEMMUL
B1EC 4C C4 B1             JMP     SPMT05
B1EF              SPMT30                  ;PV  PMT = (PV-BAL*(1+I)^-N)/((1-(1+I)^-N)/I)
                  ;                              DUE PMT = (PV-BAL*(1+I)^-N)/((1-(1+I)^-N)/I*(1+I))
B1EF 20 09 B0             JSR     Z1IMN           ;(1+I)^-N
B1F2 A9 04                LDA     #4              ;BAL
```

```
B1F4 20 FE A4                JSR     MEMMUL
B1F7 20 B6 DD                JSR     FMOVE
B1FA 20 DC AF                JSR     LDPV
B1FD 20 B9 AB                JSR     SFSUB
B200 20 9C A0                JSR     FPUSH0
B203 20 2D B0                JSR     Z11INI          ;((1-(1+I)^-N)/I)
B206 20 33 B0                JSR     ZMUL1I          ;IF DUE THEN *(1+I)
B209 20 98 AB                JSR     SPDIV
B20C 4C C4 B1                JMP     SPMT05          ;STORE NEW VALUE


B20F                 SPV                             ;PV = PRESENT VALUE
B20F AD 69 0B                LDA     ENTFLG
B212 D0 05                   BNE     SPV10
B214 A9 09           SPV05   LDA     #9
B216 4C 71 B0                JMP     MEMSTO          ;ENTER
B219                 SPV10                           ;FIND
B219 AD 68 0B                LDA     ANNFLG
B21C F0 0B                   BEQ     SPV20
                     ;                               COMPOUND INTEREST PV = FV*(1+I)^-N
B21E 20 09 B0                JSR     Z1IMN           ;(1+I)^-N
B221 A9 05                   LDA     #5              ;FV
B223 20 FE A4                JSR     MEMMUL
B226 4C 14 B2                JMP     SPV05           ;STORE NEW VALUE
B229                 SPV20                           ;PV = PMT * (1-(1+I)^-N)/I+BAL*(1+I)^-N
                     ;                               DUE = PMT * (1-(1+I)^-N)/I*(I+1)+BAL*(1+I)^-N
B229 20 2D B0                JSR     Z11INI          ;(1-(1+I)^-N)/I
B22C A9 08                   LDA     #8              ;PMT
B22E 20 FE A4                JSR     MEMMUL
B231 20 33 B0                JSR     ZMUL1I          ;IF DUE THEN *(1+I)
B234 20 9C A0                JSR     FPUSH0
B237 20 09 B0                JSR     Z1IMN           ;(1+I)^-N
B23A A9 04                   LDA     #4              ;BAL
B23C 20 FE A4                JSR     MEMMUL
B23F 20 A7 AB                JSR     SPADD
B242 4C 14 B2                JMP     SPV05           ;STORE
```

```
                        ;                               STATISTICS ROUTINES
                        ;
                        ;
                        ;                               MEM(A)<-MEM(A)+/-FRO
B245                    MEMADD
B245 20 CF A4                   JSR     MEMLD1
B248 AD 6D 0B                   LDA     MEMFLG          ;ADD?
B24B D0 06                      BNE     MEMA10          ;NO.
B24D 20 AA AB                   JSR     SFADD           ;YES
B250 4C 59 B2                   JMP     MEMA20
B253                    MEMA10
B253 20 B9 AB                   JSR     SFSUB           ;SUBTRACT
B256 20 FE A6                   JSR     SCHGSG          ;FRO-MEM(A) -> MEM(A)-FRO
B259                    MEMA20
B259 A5 CA                      LDA     DSPFLG
B25B 48                         PHA
B25C A9 01                      LDA     #1
B25E 85 CA                      STA     DSPFLG          ;DON'T STORE IN SCROLL AREA
B260 20 B6 AA                   JSR     SSTO10          ;STORE IN MEM, DISPLAY IN MEM AREA
B263 68                         PLA
B264 85 CA                      STA     DSPFLG
B266 60                         RTS

B267                    ZSIGMA                          ;MEM(A)<-MEM(A)+TOS     MEM(A+1)<-MEM(A+1)+SQU(TOS)
B267 48                         PHA
B268 20 3B A0                   JSR     FLDOS           ;FRO<-TOP OF STACK(TOS)
B26B 68                         PLA
B26C 48                         PHA                     ;RESAVE MEM #
B26D 20 45 B2                   JSR     MEMADD
B270 20 3B A0                   JSR     FLDOS
B273 20 AB AA                   JSR     SSQUAR
B276 68                         PLA
B277 18                         CLC
B278 69 01                      ADC     #1
B27A 4C 45 B2                   JMP     MEMADD

B27D                    MEMDIV                          ;FRO <- MEM(A) / N
B27D 20 C9 A4                   JSR     MEMLDO          ;MEM(A) -> FRO
B280 A9 04                      LDA     #4              ;N->FR1
B282 20 CF A4                   JSR     MEMLD1
B285 4C 9B AB                   JMP     SFDIV
```

```
B288                    SSMINU                          ;SIGMA MINUS  (DELETE PREVIOUS ENTRY)
B288 A9 01                      LDA     #1
B28A D0 02                      BNE     SIGSUB          ;JMP
B28C                    SSPLUS                          ;SIGMA PLUS: ADD NEW X,Y PAIR
B28C A9 00                      LDA     #0
B28E                    SIGSUB                          ;THIS PART IS COMMON TO BOTH SSMINU AND SSPLUS
B28E 8D 6D 0B                   STA     MEMFLG
B291 20 9C A0                   JSR     FPUSHO
B294 A9 01                      LDA     #1
B296 20 A7 A2                   JSR     PSETQ           ;N <- N+1
B299 A9 04                      LDA     #4
B29B 20 45 B2                   JSR     MEMADD

B29E A9 05                      LDA     #5              ;SIGMA X
B2A0 20 67 B2                   JSR     ZSIGMA          ;COMPUTE SIGMA X, SIGMA X SQUARED
B2A3 20 75 A0                   JSR     FPOP1           ;LOAD X
B2A6 20 3B A0                   JSR     FLDOS           ;LOAD Y (& LEAVE ON STACK)
B2A9 20 8C AB                   JSR     SFMUL
B2AC A9 09                      LDA     #9              ;SIGMA (X*Y)
B2AE 20 45 B2                   JSR     MEMADD
B2B1 A9 07                      LDA     #7              ;SIGMA Y
B2B3 20 67 B2                   JSR     ZSIGMA          ;COMPUTE SIGMA Y, SIGMA Y SQUARED
B2B6 4C 7E A0                   JMP     FPOPO           ;Y -> X

B2B9 A9 05              SXMEAN  LDA     #5              ;MEAN(X) <- SIGMA(X)/N
B2BB D0 C0                      BNE     MEMDIV          ;JMP
B2BD A9 07              SYMEAN  LDA     #7              ;MEAN(Y) <- SIGMA(Y)/N
B2BF D0 BC                      BNE     MEMDIV          ;JMP
B2C1 20 CD B2           SXSTDD  JSR     SXVARI          ;STANDARD DEVIATION (X) <- SQRT(VARIANCE(X))
B2C4 4C BA B4                   JMP     SSQRT
B2C7 20 D1 B2           SYSTDD  JSR     SYVARI          ;STDDEV(Y) <- SQRT(VAR(Y))
B2CA 4C BA B4                   JMP     SSQRT

B2CD                    SXVARI                          ;VARIANCE(X) <- (SIGMA(SQU(X))-SQU(SIGMA(X))/N)/(N+WEIGHT)
B2CD A9 05                      LDA     #5              ;SIGMA X
B2CF D0 02                      BNE     ZVAR            ;JMP
B2D1                    SYVARI                          ;VAR(Y) <- (SIGMA(SQU(Y))-SQU(SIGMA(Y))/N)/(N+WEIGHT)
B2D1 A9 07                      LDA     #7              ;SIGMA Y
B2D3                    ZVAR                            ;THIS PART IS COMMON TO BOTH SXVARI AND XYVARI
B2D3 20 E1 B2                   JSR     ZVAR2           ;COMPUTE SIGMA(SQU())-SQU(SIGMA())/N
B2D6 A9 03                      LDA     #3              ;WEIGHT
B2D8 20 CF A4                   JSR     MEMLD1
B2DB 20 AA AB                   JSR     SFADD           ;N+WEIGHT (SHOULD BE 0 OR -1)
B2DE 4C 98 AB                   JMP     SPDIV           ;NUMERATOR/(N+WEIGHT)
B2E1                    ZVAR2                           ;COMPUTE SIGMA(SQU(A))-SQU(SIGMA(A))/N
B2E1 48                         PHA                     ;SAVE REG #
B2E2 20 C9 A4                   JSR     MEMLDO          ;SIGMA
B2E5 20 AB AA                   JSR     SSQUAR          ;SQU(SIGMA)
B2E8 A9 04                      LDA     #4              ;N
B2EA 20 CF A4                   JSR     MEMLD1
B2ED 20 AA A0                   JSR     FPUSH1          ;SAVE N FOR LATER
B2F0 20 9B AB                   JSR     SFDIV
B2F3 20 B6 DD                   JSR     FMOVE
B2F6 68                         PLA                     ;RELOAD REG #
B2F7 18                         CLC                     ;AND INCREMENT
B2F8 69 01                      ADC     #1              ;SIGMA(SQU)
B2FA 20 C9 A4                   JSR     MEMLDO
B2FD 20 B9 AB                   JSR     SFSUB
B300 4C F9 AA                   JMP     SXCHGY          ;NUMERATOR <==> N   AND RETURN
```

```
B303 20 C1 B2      SCORRE  JSR     SXSTDD          ;CORRELATION = R = M * STDDEV(X)/STDDEV(Y)
B306 20 9C A0              JSR     FPUSHO
B309 20 C7 B2              JSR     SYSTDD
B30C 20 98 AB              JSR     SPDIV           ;STDDEV(X)/STDDEV(Y)
B30F 20 9C A0              JSR     FPUSHO
B312 20 5A B3              JSR     SSLOPE          ;M
B315 4C 89 AB              JMP     SPMUL

B318               SY                              ;Y <- M*X + B
B318 20 9C A0              JSR     FPUSHO          ;SAVE X
B31B 20 5A B3              JSR     SSLOPE          ;M
B31E 20 89 AB              JSR     SPMUL
B321 20 9C A0              JSR     FPUSHO
B324 20 3C B3              JSR     SYINTE          ;B = Y-INTERCEPT
B327 4C A7 AB              JMP     SPADD

B32A               SX                              ;X <- (Y-B)/M     (Y ENTERED IN X REG)
B32A 20 9C A0              JSR     FPUSHO          ;SAVE Y
B32D 20 3C B3              JSR     SYINTE          ;B
B330 20 B6 AB              JSR     SPSUB           ;Y-B
B333 20 9C A0              JSR     FPUSHO
B336 20 5A B3              JSR     SSLOPE          ;M
B339 4C 98 AB              JMP     SPDIV           ;(Y-B)/M

B33C               SYINTE                  ;Y-INTERCEPT = B = (SIGMA(Y)-M*SIGMA(X))/N
B33C 20 5A B3              JSR     SSLOPE          ;M
B33F A9 05                 LDA     #5              ;SIGMA X
B341 20 CF A4              JSR     MEMLD1
B344 20 8C AB              JSR     SFMUL           ;M*SIGMA(X)
B347 20 B6 DD              JSR     FMOVE
B34A A9 07                 LDA     #7              ;SIGMA(Y)
B34C 20 C9 A4              JSR     MEMLDO
B34F 20 B9 AB              JSR     SFSUB
B352 A9 04                 LDA     #4              ;N
B354 20 CF A4              JSR     MEMLD1
B357 4C 9B AB              JMP     SFDIV

B35A               SSLOPE                          ;SLOPE = M = (SIGMA(X*Y)-SIGMA(X)*SIGMA(Y)/N)/(SIGMA(SQU(X))-SQU(SIGMA(X))/N
B35A A9 05                 LDA     #5              ;SIGMA(X)
B35C 20 E1 B2              JSR     ZVAR2           ;COMPUTE SIGMA(SQU(X))-SQU(SIGMA(X))/N, STORE ON STACK
B35F 20 B6 DD              JSR     FMOVE           ;N-> FR1  (PUT IN FRO BY ZVAR2)
B362 A9 05                 LDA     #5              ;SIGMA(X)
B364 20 C9 A4              JSR     MEMLDO
B367 20 9B AB              JSR     SFDIV
B36A A9 07                 LDA     #7              ;SIGMA(Y)
B36C 20 CF A4              JSR     MEMLD1
B36F 20 8C AB              JSR     SFMUL
B372 20 B6 DD              JSR     FMOVE
B375 A9 09                 LDA     #9              ;SIGMA(X*Y)
B377 20 C9 A4              JSR     MEMLDO
B37A 20 B9 AB              JSR     SFSUB           ;NUMERATOR
B37D 20 75 A0              JSR     FPOP1           ;LOAD DENOMINATOR (FROM ZVAR2)
B380 4C 9B AB              JMP     SFDIV

B383               SNWEIG
B383 A9 03                 LDA     #3              ;WEIGHT FACTOR
B385 85 AF                 STA     MEMNUM
B387 4C B6 AA              JMP     SSTO10          ;MEM(3) <- X
```

```
                        ; BASIC SINE ROUTINE
                        ; TO FIX BUGS OF VERSION 5.9 OF SHEP BASIC
                        ;
                        ; BY DAVE & LARRY
                        ; 4-6-79
                        ;
                        ;
B38A                    SINMOD                          ;FIND ANGLE MOD 2*PI, 360 OR 400
B38A 20 9C A0                   JSR     FPUSHO          ;DEPENDING ON CURRENT MODE.  SAVE FR0 ON STACK FOR MOD
B38D 20 09 A5                   JSR     PIOVL           ;LOAD PI/2, 90, OR 100
B390 20 89 DD                   JSR     FLDOR
B393 A9 04                      LDA     #4
B395 20 83 AB                   JSR     INTMUL          ;MULTIPLY BY 4 (LOSE ACCURACY IN 10TH DIGIT OF 2*PI)
B398 4C 95 A8                   JMP     SMOD            ;TAKE MOD AND RETURN

                        ; COSINE ROUTINE -- ADD 90 OR PI/2 TO FR0 TO DO SIN
B39B                    SCOS
B39B 20 8A B3                   JSR     SINMOD          ;TAKE ANGLE MOD 2*PI, 360 OR 400
B39E 20 09 A5                   JSR     PIOVL           ;SET UP X & Y REGS TO LOAD PI/2 90 OR 100
B3A1 20 98 DD                   JSR     FLD1R           PUT PI/2 OR 90 INTO FR1
B3A4 20 66 DA                   JSR     FADD            FR0=FR0 + PI/2 (OR 90)
B3A7 4C AD B3                   JMP     SSIN2
                        ;
                        ;
                        ; SINE ROUTINE
                        ; COMPUTE QUADRANT, GET FRACTION AND DO POLYNOMIAL,
                        ; THEN ADJUST FOR QUADRANT
B3AA                    SSIN
B3AA 20 8A B3                   JSR     SINMOD          ;TAKE ANGLE MOD 2*PI, 360 OR 400
B3AD A5 D4              SSIN2   LDA     FR0             GET SIGN
B3AF 29 80                      AND     #$80
B3B1 85 F0                      STA     FCHRFLG         AND SAVE
B3B3 A5 D4                      LDA     FR0
B3B5 29 7F                      AND     #$7F            FR0=ABS(FR0)
B3B7 85 D4                      STA     FR0
                        ;
                        ; FR0=FR0/(PI/2) OR FR0=FR0/90
B3B9 20 09 A5                   JSR     PIOVL           ;LOAD X & Y REGS TO GET PI/2 90 OR 100
B3BC 20 98 DD                   JSR     FLD1R           FR1=PI/2 OR 90
B3BF 20 28 DB                   JSR     FDIV            FR0=FR0/FR1
B3C2 90 03                      BCC     NOSNER
B3C4                    SINERR
B3C4 4C BE AB                   JMP     CRYSND          GO IF ERROR
B3C7                    NOSNER
                        ;
                        ; IF FR0 NOW FRACTION, IT IS QUADRANT 0
                        ; ELSE, GET INTEGER OF FR0 LSD
B3C7 A9 00                      LDA     #0
B3C9 85 C5                      STA     QUADFLG         ASSUME QUADRANT 0
B3CB 38                         SEC
B3CC A5 D4                      LDA     FR0             GET EXPONENT
B3CE E9 40                      SBC     #$40            SUBTRACT 64 EXCESS
B3D0 30 39                      BMI     SINF3           GO IF QUADRANT 0
B3D2 C9 04                      CMP     #FPREC-2        IS EXPONENT TOO BIG?
B3D4 B0 EE                      BCS     SINERR          YES
                        ;
                        ; ACC=INDEX TO LSD.  GET 10*TEN'S DIGIT + ONE'S DIGIT
                        ; THEN AND WITH 3 TO GET QUADRANT
B3D6 AA                         TAX                     INDEX TO LSD IN FR0
B3D7 B5 D5                      LDA     FR0+1,X         GET LSD
```

```
B3D9 29 OF              AND     #$F             GET ONE'S DIGIT
B3DB 85 F1              STA     DIGRT           AND SAVE
B3DD B5 D5              LDA     FRO+1,X         GET LSD
B3DF 29 F0              AND     #$F0            GET TEN'S DIGIT
B3E1 4A                 LSR     A               TIMES 8
B3E2 85 C5              STA     QUADFLG         AND TEMP SAVE
B3E4 4A                 LSR     A
B3E5 4A                 LSR     A               TIMES 2
B3E6 18                 CLC
B3E7 65 C5              ADC     QUADFLG         PLUS TIMES 8 GIVES TIMES 10
B3E9 65 F1              ADC     DIGRT           PLUS ONE'S DIGIT GIVES INTEGER
B3EB 29 03              AND     #3              MASK LOW BITS
B3ED 85 C5              STA     QUADFLG         NOW HAVE QUADRANT (0,1,2, OR 3)
B3EF 86 F1              STX     DIGRT           SAVE INDEX TO LSD
                ;
                ; PUT FRO IN FR1, AND CLEAR FRACTIONAL PART OF FR1
                ; THEN GET FRO=FRACTIONAL PART OF FRO
B3F1 20 B6 DD           JSR     FMOVE           FR1=FRO
B3F4 A6 F1              LDX     DIGRT           RESTORE INDEX
B3F6 A9 00              LDA     #0
B3F8 95 E2      SINF1   STA     FR1+2,X         CLEAR FRACTIONAL PART
B3FA E8                 INX                     FROM DIGRT+1 TO END
B3FB E0 04              CPX     #FPREC-2        DONE?
B3FD 90 F9              BCC     SINF1           NO
B3FF 20 60 DA           JSR     FSUB            FRO=FRO-FR1 (FRO WILL BE FRACTIONAL PART)
                ;
                ; IF ODD QUADRANT, SET FRO=1-FRO (90 DEGREE INVERT)
B402 46 C5              LSR     QUADFLG         IS IT ODD QUADRANT?
B404 90 05              BCC     SINF3           NO
B406 A9 01              LDA     #1              ;FRO <- 1-FRO
B408 20 B0 AB           JSR     INTSUB
                ;
                ; SAVE ARG FOR LATER
B40B A2 E6      SINF3   LDX     #FPSCR          SCRATCH REG LO BYTE ADDRESS
B40D A0 05              LDY     #FPSCR/256      HI BYTE ADDRESS
B40F 20 A7 DD           JSR     FSTOR           FPSCR=FRO
                ;
                ; NOW COMPUTE SINE
                ; THIS CODE TAKEN FROM BASIC 5.9 LINES 6760-6770
B412 20 AB AA           JSR     SSQUAR          FRO=X**2
B415 B0 34              BCS     SINFIN          ;ERROR (ALREADY REPORTED)
B417 A9 06              LDA     #NSCF
B419 A2 06              LDX     #SCOEF
B41B A0 B6              LDY     #SCOEF/256
B41D 20 40 DD           JSR     PLYEVL          EVALUATE P(X**2)
B420 A2 E6              LDX     #FPSCR
B422 A0 05              LDY     #FPSCR/256
B424 20 98 DD           JSR     FLD1R           FR1=X
B427 20 DB DA           JSR     FMUL            FRO=SIN(X)=X*P(X**2)
                ;
                ; IF LOWER QUADRANT (2 OR 3) THEN FRO=-(FRO)
B42A 46 C5              LSR     QUADFLG         IS IT LOWER QUAD?
B42C 90 08              BCC     SINF4           NO
B42E A5 D4              LDA     FRO             IS FRO=0
B430 F0 0C              BEQ     SINDON          YES
B432 49 80              EOR     #$80            ELSE, FRO=-(FRO)
B434 85 D4              STA     FRO
                ;
                ; IF SIGN WAS NEGATIVE COMING IN TO ROUTINE, INVERT SIGN
                ; GOING OUT
```

```
B436 A5 D4          SINF4   LDA     FRO             ANSWER
B438 F0 04                  BEQ     SINDON          GO IF ZERO
B43A 45 F0                  EOR     FCHRFLG         INVERT ORIGINAL SIGN
B43C 85 D4                  STA     FRO             AND THIS IS END ANSWER
                    ;
                    ;
                    ; IF ABS(FRO) >= 1 THEN PERFORM PSEUDO INT(FRO)
B43E 29 7F          SINDON  AND     #$7F            WITHOUT SIGN BIT
B440 C9 40                  CMP     #$40            COMPARE $40
B442 90 07                  BCC     SINFIN
B444 18                     CLC                     NO ERROR CLEAR
B445 A9 00                  LDA     #0              STORE 0 IN LOW BYTES OF FRO
B447 85 D8                  STA     FRO+4
B449 85 D9                  STA     FRO+5
B44B 60             SINFIN  RTS
```

```
                        ;                FROM SHEPARDSON ATARI BASIC 5.9 4-5-79 (MODIFIED)
B44C                    SATAN                                    ;ARCTAN(FRO)
B44C A9 00                      LDA     #0
B44E 85 F0                      STA     FCHRFLG         ;SIGN FLAG OFF
B450 85 F1                      STA     DIGRT           ;AND TRANSFORM FLAG
B452 A5 D4                      LDA     FRO
B454 29 7F                      AND     #$7F
B456 C9 40                      CMP     #$40            ;CHECK X VS 1.0
B458 30 15                      BMI     ATAN1           ;X<1 - USE SERIES DIRECTLY
B45A A5 D4                      LDA     FRO             ;X>=1 - SAVE SIGN & TRANSFORM
B45C 29 80                      AND     #$80
B45E 85 F0                      STA     FCHRFLG         ;REMEMBER SIGN
B460 E6 F1                      INC     DIGRT
B462 A9 7F                      LDA     #$7F
B464 25 D4                      AND     FRO
B466 85 D4                      STA     FRO             ;FORCE PLUS
B468 A2 EA                      LDX     #FP9S
B46A A0 DF                      LDY     #FP9S/$100
B46C 20 95 DE                   JSR     XFORM           ;CHANGE ARG TO (X-1)/(X+1)
B46F                    ATAN1
B46F A2 E6                      LDX     #FPSCR          ;ARCTAN(X), -1<X<1 BY SERIES APPROX
B471 A0 05                      LDY     #FPSCR/256
B473 20 A7 DD                   JSR     FSTOR           ;X->FSCR
B476 20 B6 DD                   JSR     FMOVE           ;X->FR1
B479 20 DB DA                   JSR     FMUL            ;X*X->FRO
B47C B0 39                      BCS     ATNOUT          ;OVERFLOW
B47E A9 0B                      LDA     #NATCF
B480 A2 AE                      LDX     #ATCOEF
B482 A0 DF                      LDY     #ATCOEF/256
B484 20 40 DD                   JSR     PLYEVL          ;P(X*X)
B487 B0 2E                      BCS     ATNOUT          ;ERROR
B489 A2 E6                      LDX     #FPSCR
B48B A0 05                      LDY     #FPSCR/256
B48D 20 98 DD                   JSR     FLD1R           ;X->FR1
B490 20 DB DA                   JSR     FMUL            ;X*P(X*X)
B493 B0 22                      BCS     ATNOUT          ;OVERFLOW
B495 A5 F1                      LDA     DIGRT           ;WAS ARG XFORMED
B497 F0 10                      BEQ     ATAN2           ;NO.
B499 A2 F0                      LDX     #PIOV4          ;YES-ADD ARCTAN(1) = PI/4
B49B A0 DF                      LDY     #PIOV4/256
B49D 20 98 DD                   JSR     FLD1R
B4A0 20 66 DA                   JSR     FADD
B4A3 A5 F0                      LDA     FCHRFLG         ;GET ORG SIGN
B4A5 05 D4                      ORA     FRO
B4A7 85 D4                      STA     FRO             ;ATAN(-X) = -ATAN(X)
B4A9                    ATAN2
B4A9 A5 FB                      LDA     RADFLG          ;RAD OR DEG
B4AB F0 0A                      BEQ     ATNOUT          ;RAD - FINI
                        ;       CLC
                        ;       ADC     #PIOV18-FPREC
                        ;       TAX                     ;DIVIDE BY PI/180 OR PI/200
B4AD A2 36                      LDX     #PIOV18         ;ABOVE IS USED IF GRADS ALLOWED
B4AF A0 B6                      LDY     #PIOV18/256
B4B1 20 98 DD                   JSR     FLD1R
B4B4 4C 9B AB                   JMP     SFDIV
B4B7                    ATNOUT
B4B7 4C BC AB                   JMP     CRYCHK
```

```
                        ;               FROM SHEPARDSON ATARI BASIC 5.9 4-5-79 (MODIFIED)
                        ;               USES NEWTON-RAPHSON ITERATION
                        ;               F(Y) = Y*Y - X
                        ;               FPRIME(Y) = 2*Y
                        ;               Y[I+1] = Y[I] - F(Y[I]) / FPRIME(Y[I]) = Y[I] + .5*((X/Y[I])-Y[I])
B4BA                    SSQRT                           ;X<-SQRT(X)
B4BA A9 00                      LDA     #0
B4BC 85 F1                      STA     DIGRT
B4BE A5 D4                      LDA     FR0
B4C0 10 08                      BPL     SQR0
B4C2 A9 76                      LDA     #BITMSG
B4C4 20 E5 9B                   JSR     ERRSUB          ;<0 => ERROR
B4C7 20 B9 A5                   JSR     SABSVA          ;TAKE ABS VALUE AND DO SQUARE ROOT (ABSVAL LOADS FR0 INTO A)
B4CA                    SQR0
B4CA C9 3F                      CMP     #$3F
B4CC F0 17                      BEQ     FSQR            ;X IN RANGE OF APPROX - GO DO IT TO IT
B4CE 18                         CLC
B4CF 69 01                      ADC     #1
B4D1 85 F1                      STA     DIGRT           ;NOT IN RANGE - TRANSFORM
B4D3 85 E0                      STA     FR1             ;MANTISSA = 1
B4D5 A9 01                      LDA     #1
B4D7 85 E1                      STA     FR1+1
B4D9 A2 03                      LDX     #FPREC-3                ;CHANGED 5/11/79 FROM FPREC-2
B4DB A9 00                      LDA     #0
B4DD                    SQR1
B4DD 95 E2                      STA     FR1+2,X
B4DF CA                         DEX
B4E0 10 FB                      BPL     SQR1
B4E2 20 28 DB                   JSR     FDIV            ;X/100**N
B4E5                    FSQR                            ;SQR(X) 0.1<=X<1
B4E5 A9 06                      LDA     #6
B4E7 85 EF                      STA     ESIGN
B4E9 A2 E6                      LDX     #FSCR
B4EB A0 05                      LDY     #FSCR/256
B4ED 20 A7 DD                   JSR     FSTOR           ;STASH X IN FSCR
B4F0 A9 02                      LDA     #2
B4F2 20 B0 AB                   JSR     INTSUB          ;2-X
B4F5 A2 E6                      LDX     #FSCR
B4F7 A0 05                      LDY     #FSCR/256
B4F9 20 98 DD                   JSR     FLD1R           ;X->FR1
B4FC 20 DB DA                   JSR     FMUL            ;X*(2-X) : 1ST APPROX
B4FF                    SQRLP
B4FF A2 EC                      LDX     #FSCR1
B501 A0 05                      LDY     #FSCR1/256
B503 20 A7 DD                   JSR     FSTOR           ;Y->FSCR1
B506 20 B6 DD                   JSR     FMOVE           ;Y->FR1
B509 A2 E6                      LDX     #FSCR
B50B A0 05                      LDY     #FSCR/256
B50D 20 89 DD                   JSR     FLDOR
B510 20 28 DB                   JSR     FDIV            ;X/Y
B513 A2 EC                      LDX     #FSCR1
B515 A0 05                      LDY     #FSCR1/256
B517 20 98 DD                   JSR     FLD1R
B51A 20 60 DA                   JSR     FSUB            ;(X/Y)-Y
B51D A2 6C                      LDX     #FHALF
B51F A0 DF                      LDY     #FHALF/256
B521 20 98 DD                   JSR     FLD1R
B524 20 DB DA                   JSR     FMUL            ;.5*((X/Y)-Y)=DELTAY
B527 A5 D4                      LDA     FR0             ;DELTA 0
B529 F0 0E                      BEQ     SQRDON
```

```
B52B A2 EC                      LDX     #FSCR1
B52D A0 05                      LDY     #FSCR1/256
B52F 20 98 DD                   JSR     FLD1R
B532 20 66 DA                   JSR     FADD            ;Y=Y+DELTA Y
B535 C6 EF                      DEC     ESIGN           ;COUNT & LOOP
B537 10 C6                      BPL     SQRLP
B539                    SQRDON
B539 A2 EC                      LDX     #FSCR1          ;DELTA = 0 - GET Y BACK
B53B A0 05                      LDY     #FSCR1/256
B53D 20 89 DD                   JSR     FLDOR
                        ;                               WAS ARG TRANSFORMED?
B540 A5 F1                      LDA     DIGRT
B542 F0 20                      BEQ     SQROUT          ;NO FINI
B544 38                         SEC
B545 E9 40                      SBC     #$40
                        ;                               ;YES - TRANSFORM RESULT TO MATCH
B547 4A                         LSR     A               ;DIVIDE EXP BY 2
B548 18                         CLC
B549 69 40                      ADC     #$40
B54B 85 E0                      STA     FR1
B54D A5 F1                      LDA     DIGRT
B54F 6A                         ROR     A
B550 A9 01                      LDA     #1              ;MANTISSA = 1
B552 90 02                      BCC     SQR2            ;WAS EXP ODD OR EVEN
B554 A9 10                      LDA     #$10            ;ODD - MANT = 10
B556                    SQR2
B556 85 E1                      STA     FR1+1
B558 A2 03                      LDX     #FPREC-3                ;CHANGED 5/11/79 FROM FPREC-2
B55A A9 00                      LDA     #0
B55C                    SQR3
B55C 95 E2                      STA     FR1+2,X         ;CLEAR REST OF MANTISSA
B55E CA                         DEX
B55F 10 FB                      BPL     SQR3
B561 20 DB DA                   JSR     FMUL            ;SQR(X) = SQR(X/100**N) * (10**N)
B564                    SQROUT
B564 20 B9 A5                   JSR     SABSVA
B567 60                         RTS
```

```
                        ;                       DATA
                        ;               THE FOLLOWING TABLES MUST NOT CROSS PAGE BOUNDARIES
                                *=*-1/256+1*256
                        ;               MY EXPERIMENTAL SCOEF FOR SIN,COS   (9 TERMS INSTEAD OF 6)
                        ;EXPSC
                        ;       .BYTE   $3A,$06,$06,$69,$35,$73 ;6.066935731E-12
                        ;       .BYTE   $BB,$06,$68,$80,$35,$12 ;-6.688035123E-10 = -(PI/2)^15/15!
                        ;       .BYTE   $3C,$05,$69,$21,$72,$92 ;5.692172922E-08 = (PI/2)^13/13!
                        ;       .BYTE   $BD,$03,$59,$88,$43,$24 ;-.00000359884324 - (PI/2)^11/11!
                        ;       .BYTE   $3E,$01,$60,$44,$11,$85 ;0.000160441185
                        ;       .BYTE   $BE,$46,$81,$75,$41,$35 ;-.004681754155
                        ;       .BYTE   $3F,$07,$96,$92,$62,$62 ;0.0796926262
                        ;       .BYTE   $BF,$64,$59,$64,$09,$75 ;-.6459640975
                        ;       .BYTE   $40,$01,$57,$07,$96,$32 ;PI/2 = 1.570796327

B600 40 03 14           PICONST .BYTE   $40,$03,$14,$15,$92,$65 ;PI = 3.14159265
B603 15 92 65
B606                    SCOEF
B606 BD 03 54                   .BYTE   $BD,$03,$54,$14,$99,$39 ;-.00000354149939
B609 14 99 39
B60C 3E 01 60                   .BYTE   $3E,$01,$60,$44,$27,$52 ;0.000160442752
B60F 44 27 52
B612 BE 46 81                   .BYTE   $BE,$46,$81,$75,$43,$55 ;-.004681754355
B615 75 43 55
B618 3F 07 96                   .BYTE   $3F,$07,$96,$92,$62,$39 ;0.0796926239
B61B 92 62 39
B61E BF 64 59                   .BYTE   $BF,$64,$59,$64,$08,$67 ;-.6459640867
B621 64 08 67
B624 40 01 57           RADPI2  .BYTE   $40,$01,$57,$07,$96,$32 ;PI/2 = 1.570796327
B627 07 96 32
B62A 40 90 00                   .BYTE   $40,$90,0,0,0,0         ;90 (DEGREES)
B62D 00 00 00
                        ;       .BYTE   $41,$01,0,0,0,0         ;100 (GRADS)
B630 42 06 55           C65536  .BYTE   $42,$06,$55,$36,0,0     ;65536 IN FP (USED IN BINFP)
B633 36 00 00
B636 3F 01 74           PIOV18  .BYTE   $3F,$01,$74,$53,$29,$25 ;PI/180 = .0174532925
B639 53 29 25
                        ;       .BYTE   $3F,$01,$57,$07,$96,$33 ;PI/200 = .0157079633
                        ;C10000 .BYTE   $42,$01,0,0,0,0         ;10000 (USED IN DAY CALCULATIONS)
                        ;C365   .BYTE   $41,$03,$65,0,0,0       ;365
                        ;CPT75  .BYTE   $3F,$75,0,0,0,0         ;.75 = 3/4
B63C 40 01 80           C1PT8   .BYTE   $40,$01,$80,0,0,0       ;1.8 (USED IN SCELSI)
B63F 00 00 00
B642                    LENGTH
B642 40 01 00                   .BYTE   $40,$01,0,0,0,0         ;M -> M     = 1      EXACTLY
B645 00 00 00
B648 3F 02 54                   .BYTE   $3F,$02,$54,0,0,0       ;INCHES->M = .0254 EXACTLY
B64B 00 00 00
B64E 3F 30 48                   .BYTE   $3F,$30,$48,0,0,0       ;FEET         .3048    "
B651 00 00 00
B654 3F 91 44                   .BYTE   $3F,$91,$44,0,0,0       ;YARDS        .9144    "
B657 00 00 00
B65A 41 16 09                   .BYTE   $41,$16,$09,$34,$40,0   ;MILES   1609.344     "
B65D 34 40 00
B660 3F 01 00                   .BYTE   $3F,$01,0,0,0,0         ;CM           .01      "
B663 00 00 00
B666 41 10 00                   .BYTE   $41,$10,0,0,0,0         ;KM      1000          "
B669 00 00 00
```

```
B66C 41 18 52                .BYTE   $41,$18,$52,0,0,$04     ;NAUTMI  1852.000004 ????????
B66F 00 00 04

B672                    MASS
B672 3F 02 83                .BYTE   $3F,$02,$83,$49,$52,$31 ;OZ->KG = .02834952313 (NOT EXACT)
B675 49 52 31
B678 3F 45 35                .BYTE   $3F,$45,$35,$92,$37,0   ;LB       .45359237     ??
B67B 92 37 00
B67E 3E 10 00                .BYTE   $3E,$10,0,0,0,0         ;GM       .001        EXACTLY
B681 00 00 00

B684                    VOLUME
                        ;       .BYTE   $3F,$16,$66,$66,$66,$67 ;TSP->FLOZ = .1666666667
                        ;       .BYTE   $3F,$50,0,0,0,0         ;TBSP        .5      EXACTLY
                        ;       .BYTE   $40,8,0,0,0,0           ;CUPS        8         "
                        ;       .BYTE   $40,$32,0,0,0,0         ;QUARTS     32         "
B684 41 01 28                .BYTE   $41,$01,$28,0,0,0       ;GAL       128         "
B687 00 00 00
B68A 40 33 81                .BYTE   $40,$33,$81,$40,$22,$66 ;LITERS    33.81402266 NOT EXACT?
B68D 40 22 66
B690                    DEGREE
B690 3F 90 00                .BYTE   $3F,$90,0,0,0,0         ;180/200 = .9 GRAD -> DEG
B693 00 00 00
B696 40 57 29                .BYTE   $40,$57,$29,$57,$79,$51 ;180/PI = 57.29577951 RAD -> DEG
B699 57 79 51

                        ;                       MY EXPERIMENTAL P10COF FOR EXP FUNCTION
                        ;P10COF
                        ;       .BYTE   $3D,$09,$79,$28,$29,$75 ;.000009792829753 = (LN(10)/2)^9/9!
                        ;       .BYTE   $3D,$76,$55,$34,$94,$63 ;.00007655349463  = (LN(10)/2)^8/8!
                        ;       .BYTE   $3E,$05,$31,$94,$81,$65 ;.000531948165
                        ;       .BYTE   $3E,$32,$34,$31,$01,$36 ;.003234310136
                        ;       .BYTE   $3F,$01,$68,$55,$71,$65 ;.0168557165
                        ;       .BYTE   $3F,$07,$32,$03,$44,$68 ;.0732034468       = (LN(10)/2)^4/4!
                        ;       .BYTE   $3F,$25,$43,$34,$82,$44 ;.2543348244
                        ;       .BYTE   $3F,$66,$27,$37,$26,$38 ;.6627372638       = (LN(10)/2)^2/2!
                        ;       .BYTE   $40,$01,$15,$12,$92,$55 ;1.15129255        = LN(10)/2
                        ;       .BYTE   $3F,$99,$99,$99,$99,$99 ;.999999999          APPROX. 1
                        ;
                        ;                       LENGTH OF EACH MONTH + 1 IN BCD
                        ;MAXDAY .BYTE   1+$31,$30,1+$31,1+$30,1+$31,1+$30,1+$31,1+$31,1+$30,1+$31,1+$30,1+$31
                        ;DAYTRM .BYTE   0,0,3,3,4,4,5,5,5,6,6,7 ;# OF DAYS LESS THAN 31/MONTH FOR EACH MONTH
                        ;DAYTBL .BYTE   "SATSUNMONTUEWEDTHUFRI"
                        ;                               SPECIAL SINGLE CHAR COMMANDS
B69C 2A 2F 2B           TOKCHR  .BYTE   "*/+-()=^!%",UPAROW,DNAROW,LFAROW,RTAROW
B69F 2D 28 29
B6A2 3D 5E 21
B6A5 25 1C 1D
B6A8 1E 1F
B6AA                    TOKEND
                        ;                               TOKEN NUMBERS FOR TOKCHR COMMANDS
B6AA 8E 8F 90           TOKTBL  .BYTE   STAR,SLASH,PLUS,MINUS,LPAR,RPAR,EQUAL,POWER,FACTOR,PERCEN
B6AD 91 92 93
B6B0 94 5B 28
B6B3 54
                        ;                       BSTEP,SSTEP,DELETE,INSERT ARE PART OF BOTH TOKTBL & SPCTBL
                        ;                               SPECIAL COMMANDS IN STORE PROGRAM MODE (EXECUTED IMMEDIATELY, NOT STORED)
B6B4 0C 76 21           SPCTBL  .BYTE   BSTEP,SSTEP,DELETE,INSERT,CLP,ZEND,PROGRAM
B6B7 39 00 24
B6BA 5E
```

COLLEEN CALCULATOR, BY C SHAW

```
B6BB                    SPCEND
B6BB 50 3A 9B           PBUFF    .BYTE   "P:",CR
B6BE 4B 3A 9B           KBUFF    .BYTE   "K:",CR
B6C1 45 52 52           ERRMSG   .BYTE   "ERROR -"
B6C4 4F 52 20
B6C7 2D
B6C8 20 2A 2A           STARMS   .BYTE   " ***"
B6CB 2A
                        ;GRAPHICS CHARS FOR SCREEN DISPLAY      -64 => CONTROL KEY HIT (USED IN PTLIN1)
B6CC 11 17 05           CHRTAB   .BYTE   'Q-64,'W-64,'E-64,'A-64,'S-64,'D-64,'Z-64,'X-64,'C-64
B6CF 01 13 04
B6D2 1A 18 03
B6D5 58 59 32           CHTAB2   .BYTE   "XY23456789"                                    STACK LABELS
B6D8 33 34 35
B6DB 36 37 38
B6DE 39
```

```
                        ;       OUTPUT FROM BASIC PROGRAM DK1:WORDSG.BAS
B6DF 20 45 52           TABLE   .BYTE   " ERTANOSICLDMPUGFXYVHBKWQZJ"
B6E2 54 41 4E
B6E5 4F 53 49
B6E8 43 4C 44
B6EB 4D 50 55
B6EE 47 46 58
B6F1 59 56 48
B6F4 42 4B 57
B6F7 51 5A 4A
                        *=*-1/256+1*256 ;GOTO NEXT PAGE BOUNDARY
B700                    ERRTBL
                        ;               TWO OPERATORS IN A ROW
B700 18 40 87           TOPMSG  .BYTE    24,64,135,23,226,53,71,56,25,97,81,55,8
B703 17 E2 35
B706 47 38 19
B709 61 51 37
B70C 08
                        ;               NOT VALID COMMAND
B70D 12 67 41           KEYMSG  .BYTE    18,103,65,4,91,156,26,125,213,108
B710 04 5B 9C
B713 1A 7D D5
B716 6C
                        ;               CHARACTER NOT VALID IN THIS BASE
B717 24 A0 55           NOMMSG  .BYTE    36,160,85,53,164,35,22,116,16,69,185,193,150,20,5,152,16,101,130
B71A 35 A4 23
B71D 16 74 10
B720 45 B9 C1
B723 96 14 05
B726 98 10 65
B729 82
                        ;               NO DIGIT IN EXPONENT
B72A 16 67 1C           NODMSG  .BYTE    22,103,28,144,9,65,150,18,2,231,98,100
B72D 90 09 41
B730 96 12 02
B733 E7 62 64
                        ;               BINARY HEX OCTAL OVERFLOW
B736 20 06 96           BOMSG   .BYTE    32,6,150,83,3,16,82,2,23,164,91,23,4,35,1,183,8
B739 53 03 10
B73C 52 02 17
B73F A4 5B 17
B742 04 23 01
B745 B7 08
                        ;               NUMBER STACK EMPTY
B747 15 6F D0           NSEMSG  .BYTE    21,111,208,98,49,132,90,7,18,222,64,48
B74A 62 31 84
B74D 5A 07 12
B750 DE 40 30
                        ;               NUMBER STACK FULL
B753 14 6F D0           NSFMSG  .BYTE    20,111,208,98,49,132,90,7,16,31,187
B756 62 31 84
B759 5A 07 10
B75C 1F BB
                        ;               OPERATOR STACK EMPTY
B75E 16 7E 23           OSEMSG  .BYTE    22,126,35,84,115,24,69,160,113,45,228,3
B761 54 73 18
B764 45 A0 71
B767 2D E4 03
                        ;               OPERATOR STACK FULL
B76A 15 7E 23           OSFMSG  .BYTE    21,126,35,84,115,24,69,160,113,1,251,176
```

```
B76D 54 73 18
B770 45 A0 71
B773 01 FB B0
                        ;                       NUMBER OUT OF RANGE
B776 16 6F D0   BITMSG  .BYTE    22,111,208,98,49,127,65,112,17,53,96,2
B779 62 31 7F
B77C 41 70 11
B77F 35 60 02
                        ;                       TOO MANY CHARACTERS
B782 15 47 71   DIGMSG  .BYTE    21,71,113,213,96,49,160,85,53,164,35,128
B785 D5 60 31
B788 A0 55 35
B78B A4 23 80
                        ;                       PRINTER DOES NOT RESPOND
B78E 18 E3 96   NOPMSG  .BYTE    24,227,150,66,49,199,40,22,116,19,40,231,108
B791 42 31 C7
B794 28 16 74
B797 13 28 E7
B79A 6C
                        ;                       ARITHMETIC OVERFLOW
B79B 17 53 94   CRYMSG  .BYTE    23,83,148,5,210,73,161,112,66,48,27,112,128
B79E 05 D2 49
B7A1 A1 70 42
B7A4 30 1B 70
B7A7 80
                        ;                       BIN REQUIRES BITS 16 OR LESS
B7A8 25 06 96   BIMSG   .BYTE    37,6,150,19,32,159,147,40,16,105,72,16,243,16,243,97,115,27,40,128
B7AB 13 20 9F
B7AE 93 28 10
B7B1 69 48 10
B7B4 F3 10 F3
B7B7 61 73 1B
B7BA 28 80
                        ;                       END OF PROGRAM MEMORY
B7BC 18 26 C1   EPMSG   .BYTE    24,38,193,112,17,227,112,3,93,29,45,115,3
B7BF 70 11 E3
B7C2 70 03 5D
B7C5 1D 2D 73
B7C8 03
                        ;                       UNIT MISMATCH
B7C9 0E F6 94   UNIMSG  .BYTE    14,246,148,29,152,213,74,5
B7CC 1D 98 D5
B7CF 4A 05
                        ;                       CALL STACK EMPTY
B7D1 12 A5 BB   CLEMSG  .BYTE    18,165,187,24,69,160,113,45,228,3
B7D4 18 45 A0
B7D7 71 2D E4
B7DA 03
                        ;                       CALL STACK FULL
B7DB 11 A5 BB   CLFMSG  .BYTE    17,165,187,24,69,160,113,1,251,176
B7DE 18 45 A0
B7E1 71 01 FB
B7E4 B0
                        *=*-1/256+1*256         ;GOTO NEXT PAGE BOUNDARY
                        ;                       ENTER PROGRAM MEMORY ADDRESS 0-1023
B800 37 26 42   PROMSG  .BYTE    55,38,66,49,227,112,3,93,29,45,115,3,21,204,50,136,16,243,0,242,208,243,16,243,0,243,32,243
B803 31 E3 70
B806 03 5D 1D
B809 2D 73 03
B80C 15 CC 32
```

COLLEEN CALCULATOR, BY C SHAW

```
B80F 88 10 F3
B812 00 F2 D0
B815 F3 10 F3
B818 00 F3 20
B81B F3 30
                        ;                   ENTER 0-7 OR 9 (FLOATING DECIMAL)
B81D 35 26 42       FIXMSG  .BYTE     53,38,66,49,15,48,15,45,15,55,23,49,15,57,16,242,128,27,117,73,96,1,194,169,213,176,242,144
B820 31 0F 30
B823 0F 2D 0F
B826 37 17 31
B829 0F 39 10
B82C F2 80 1B
B82F 75 49 60
B832 01 C2 A9
B835 D5 B0 F2
B838 90
                        ;                   ENTER 1-32
B839 16 26 42       BTSMSG  .BYTE     22,38,66,49,15,49,15,45,15,51,15,50
B83C 31 0F 31
B83F 0F 2D 0F
B842 33 0F 32
                        ;                   ENTER MEMORY REGISTER 0-99
B845 28 26 42       MEMMSG  .BYTE     40,38,66,49,210,215,48,49,50,0,152,66,49,15,48,15,45,15,57,15,57
B848 31 D2 D7
B84B 30 31 32
B84E 00 98 42
B851 31 0F 30
B854 0F 2D 0F
B857 39 0F 39
                        ;                   ->FAHRENHEIT
B85A 15 0F 2D       CELMSG  .BYTE     21,15,45,15,62,1,80,83,38,5,41,64
B85D 0F 3E 01
B860 50 53 26
B863 05 29 40
                        ;                   ->CELSIUS
B866 0F 0F 2D       FAHMSG  .BYTE     15,15,45,15,62,162,184,159,128
B869 0F 3E A2
B86C B8 9F 80
                        ;                   INTERMEDIATE UNITS:
B86F 16 96 42       CN1MSG  .BYTE     22,150,66,61,44,149,66,31,105,72,15,58
B872 3D 2C 95
B875 42 1F 69
B878 48 0F 3A
                        ;                   ENTER DESIRED UNITS
B87B 13 26 42       CN2MSG  .BYTE     19,38,66,49,194,137,50,193,246,148,128
B87E 31 C2 89
B881 32 C1 F6
B884 94 80
                        ;                   CONVERSION COMPLETE
B886 14 A7 60       CN3MSG  .BYTE     20,167,96,66,56,151,97,167,222,178,66
B889 42 38 97
B88C 61 A7 DE
B88F B2 42
                        ;                   ->DECIMAL DEGREES
B891 18 0F 2D       ZDMS    .BYTE     24,15,45,15,62,194,169,213,177,194,0,50,40
B894 0F 3E C2
B897 A9 D5 B1
B89A C2 00 32
B89D 28
                        ;                   ->DMS
```

```
B89E 0B 0F 2D       ZDCDEG  .BYTE    11,15,45,15,62,205,128
B8A1 0F 3E CD
B8A4 80
                    ;                       ->POLAR Y,X->Y=ANGLE,X=RADIUS
B8A5 3A 0F 2D       ZRECT   .BYTE    58,15,45,15,62,231,181,49,3,15,44,2,15,45,15,62,3,15,61,86,0,178,15,44,2,15,61,53,201,248
B8A8 0F 3E E7
B8AB B5 31 03
B8AE 0F 2C 02
B8B1 0F 2D 0F
B8B4 3E 03 0F
B8B7 3D 56 00
B8BA B2 0F 2C
B8BD 02 0F 3D
B8C0 35 C9 F8
                    ;                       ->RECTANGULAR Y=ANGLE,X=RADIUS->Y,X
B8C3 41 0F 2D       ZPOLAR  .BYTE    65,15,45,15,62,50,164,86,0,251,83,16,48,243,213,96,11,32,242,192,32,243,211,92,159,128,242,
B8C6 0F 3E 32
B8C9 A4 56 00
B8CC FB 53 10
B8CF 30 F3 D5
B8D2 60 0B 20
B8D5 F2 C0 20
B8D8 F3 D3 5C
B8DB 9F 80 F2
B8DE D0 F3 E0
                    ;                                                       ->RECTANGULAR Y=ANGLE,X=RADIUS->Y,X      180
B8E1 30 F2 C0               .BYTE           48,242,192,32
B8E4 20
                    ;                       METERS
B8E5 06 D2 42       ZM      .BYTE    6,210,66,56
B8E8 38
                    ;                       KILOGRAMS
B8E9 0B 07 9B       ZKG     .BYTE    11,7,155,112,3,93,128
B8EC 70 03 5D
B8EF 80
                    ;                       FLUID OUNCES
B8F0 0D 01 BF       ZFL     .BYTE    13,1,191,156,23,246,162,128
B8F3 9C 17 F6
B8F6 A2 80
                    ;                       DEGREES
B8F8 08 C2 00       ZDEG    .BYTE    8,194,0,50,40
B8FB 32 28
                    ;                       [    ATARI CALCULATOR COPYRIGHT 1979  ]     ALG  RAD  DEC BITS16 FIX9 OFF
B8FD 71 0F 7D       STATLN  .BYTE    113,15,125,17,17,84,83,145,165,186,251,84,115,26,126,3,57,0,5,65,15,49,15,57,15,55,15,57,17
B900 11 11 54
B903 53 91 A5
B906 BA FB 54
B909 73 1A 7E
B90C 03 39 00
B90F 05 41 0F
B912 31 0F 39
B915 0F 37 0F
B918 39 11 0F
                    ;                                                       [    ATARI CALCULATOR COPYRIGHT 1979  ]     ALG  RAD
B91B 9B 11 11               .BYTE           155,17,17,21,176,1,19,92,17,194,161,6,148,128,243,16,243,97,1,144,32,243,145,112,16,1
B91E 15 B0 01
B921 13 5C 11
B924 C2 A1 06
B927 94 80 F3
B92A 10 F3 61
```

```
B92D 01 90 20
B930 F3 91 70
B933 10 11 11
B936 10
                        ;                 |      STACK       |REG       CONTENTS|
B937 31 0F 7C           STKLIN  .BYTE     49,15,124,17,17,17,132,90,7,17,17,17,15,124,50,0,17,17,17,26,118,66,100,128,247,192
B93A 11 11 11
B93D 84 5A 07
B940 11 11 11
B943 0F 7C 32
B946 00 11 11
B949 11 1A 76
B94C 42 64 80
B94F F7 C0
                        ;                 ENTER FILE SPEC (NO QUOTES)
B951 23 26 42           FSPMSG  .BYTE     35,38,66,49,1,155,33,142,42,16,242,134,113,9,247,66,128,242,144
B954 31 01 9B
B957 21 8E 2A
B95A 10 F2 86
B95D 71 09 F7
B960 42 80 F2
B963 90
B964                    KEYWRD

                        ;                                                 ABSVAL      0
B964 85 06 80                     .BYTE         133,6,128,69
B967 45
                        ;                                                 ACOS        1
B968 B4 5A 78                     .BYTE         180,90,120
                        ;                                                 ADV         2
B96B 45 C0                        .BYTE         69,192
                        ;                                                 ALG         3
B96D 44 5B 00                     .BYTE         68,91,0
                        ;                                                 ALGN        4
B970 55 B0 06                     .BYTE         85,176,6
                        ;                                                 AND         5
B973 35 6C                        .BYTE         53,108
                        ;                                                 ANNUITY     6
B975 85 66 F9                     .BYTE         133,102,249,64
B978 40
                        ;                                                 ASIN        7
B979 34 58 96                     .BYTE         52,88,150
                        ;                                                 ATAN        8
B97C 45 45                        .BYTE         69,69
                        ;                                                 BAL         9
B97E 64 06 5B                     .BYTE         100,6,91
                        ;                                                 BIN         10
B981 40 69                        .BYTE         64,105
                        ;                                                 BITS        11
B983 65 06 94                     .BYTE         101,6,148
                        ;                                                 BSTEP       12
B986 86 06 84                     .BYTE         134,6,132,46
B989 2E
                        ;                                                 CALL        13
B98A 4A 5B                        .BYTE         74,91
                        ;                                                 CDEG        14
B98C B5 AC 20                     .BYTE         181,172,32
                        ;                                                 CELSIUS     15
B98F 07 A2 B8                     .BYTE         7,162,184,159
B992 9F
```

```
                        ;
B993 86 A0 03                   .BYTE       134,160,3,92
B996 5C
                        ;
B997 9A 05 00                   .BYTE       154,5,0,128,6
B99A 80 06
                        ;
B99C 6A B9 69                   .BYTE       106,185,105
                        ;
B99F 45 AB D2                   .BYTE       69,171,210
                        ;
B9A2 D7 AB E3                   .BYTE       215,171,227,112
B9A5 70
                        ;
B9A6 03 AB                      .BYTE       3,171
                        ;
B9A8 34 AB 02                   .BYTE       52,171,2
                        ;
B9AB 2A                         .BYTE       42
                        ;
B9AC DB AD E6                   .BYTE       219,173,230,201,100,56
B9AF C9 64 38
                        ;
B9B2 4A A7 DE                   .BYTE       74,167,222,178,210,100
B9B5 B2 D2 64
                        ;
B9B8 8A 76 49                   .BYTE       138,118,73,111
B9BB 6F
                        ;
B9BC 2B A7 33                   .BYTE       43,167,51,43,84,151
B9BF 2B 54 97
                        ;
B9C2 63 A7                      .BYTE       99,167
                        ;
B9C4 84 A3 5C                   .BYTE       132,163,92
                        ;
B9C7 6C AC 20                   .BYTE       108,172,32
                        ;
B9CA 03 C2                      .BYTE       3,194
                        ;
B9CC A4 C2 00                   .BYTE       164,194,0
                        ;
B9CF 6C 2B 24                   .BYTE       108,43,36
                        ;
B9D2 24 C9 04                   .BYTE       36,201,4
                        ;
B9D5 3C D8                      .BYTE       60,216
                        ;
B9D7 32 6C                      .BYTE       50,108
                        ;
B9D9 52 64 23                   .BYTE       82,100,35
                        ;
B9DC 52 02 E2                   .BYTE       82,2,226
                        ;
B9DF 72 02 E4                   .BYTE       114,2,228,38
B9E2 26
                        ;
B9E3 A0 15 A4                   .BYTE       160,21,164,115,149
B9E6 73 95
                        ;
```

```
CGRAD           16

CHGSGN          17

CLINIT          18

CLMEM           19

CLPROG          20

CLR             21

CLX             22

CM              23

CMPNDINTRST     24

COMPLEMENT      25

CONTINUE        26

CORRELATION     27

COS             28

CRAD            29

DCDEG           30

DEC             31

DEG             32

DELETE          33

DIV             34

DMS             35

END             36

ENTER           37

EXPE            38

EXPTEN          39

FACTORIAL       40

FAHRENHEIT      41
```

```
B9E8 BD 01 50          .BYTE   189,1,80,83,38,5,41
B9EB 53 26 05
B9EE 29
                ;
B9EF 45 01 22          .BYTE   69,1,34
                ;
B9F2 45 01 96          .BYTE   69,1,150
                ;
B9F5 C5 01 90          .BYTE   197,1,144
                ;
B9F8 26 01 B7          .BYTE   38,1,183,10
B9FB 0A
                ;
B9FC 90 13 5A          .BYTE   144,19,90,73,118
B9FF 49 76
                ;
BA01 40 10             .BYTE   64,16
                ;
BA03 47 01 04          .BYTE   71,1,4,207
BA06 CF
                ;
BA07 27 01 04          .BYTE   39,1,4,115
BA0A 73
                ;
BA0B C4 00 5B          .BYTE   196,0,91
                ;
BA0E 30 0D             .BYTE   48,13
                ;
BA10 50 07 47          .BYTE   80,7,71
                ;
BA13 60 03 5D          .BYTE   96,3,93
                ;
BA16 85 05 20          .BYTE   133,5,32
                ;
BA19 21                .BYTE   33
                ;
BA1A 97 96 A0          .BYTE   151,150,160,82
BA1D 52
                ;
BA1E 86 96 82          .BYTE   134,150,130,52
BA21 34
                ;
BA22 89 64 20          .BYTE   137,100,32,2
BA25 02
                ;
BA26 36 96 4D          .BYTE   54,150,77,44
BA29 2C
                ;
BA2A 40 70             .BYTE   64,112
                ;
BA2C 03 07             .BYTE   3,7
                ;
BA2E D3 B0             .BYTE   211,176
                ;
BA30 64 B9 84          .BYTE   100,185,132
                ;
BA33 6B 94 23          .BYTE   107,148,35
                ;
BA36 82 B6             .BYTE   130,182
                ;
```

```
FEET        42
FIND        43
FIX         44
FLOZ        45
FRACTION    46
FV          47
FVDUE       48
FVORD       49
GAL         50
GM          51
GOTO        52
GRAMS       53
HEX         54
I           55
INCHES      56
INSERT      57
INTEGER     58
INTMED      59
KG          60
KM          61
LB          62
LIST        63
LITERS      64
LN          65
LOGTEN      66
```


COLLEEN CALCULATOR, BY C SHAW

```
BA38 7B 70 04          .BYTE   123,112,4,38
BA3B 26
                ;
BA3C 6B 80 50          .BYTE   107,128,80
                ;
BA3F 16 D2 42          .BYTE   22,210,66,56
BA42 38
                ;
BA43 5D 9B 28          .BYTE   93,155,40
                ;
BA46 5D B7 5C          .BYTE   93,183,92
                ;
BA49 3D 7C             .BYTE   61,124
                ;
BA4B 6D 85 04          .BYTE   109,133,4
                ;
BA4E 21                .BYTE   33
                ;
BA4F 6D 65 F4          .BYTE   109,101,244,154,91,217,178
BA52 9A 5B D9
BA55 B2
                ;
BA56 83 67             .BYTE   131,103
                ;
BA58 E7 67 43          .BYTE   231,103,67,90
BA5B 5A
                ;
BA5C 2A 60 82          .BYTE   42,96,130,144,0,84
BA5F 90 00 54
                ;
BA62 37 A4             .BYTE   55,164
                ;
BA64 57 01 01          .BYTE   87,1,1
                ;
BA67 27                .BYTE   39
                ;
BA68 62 73             .BYTE   98,115
                ;
BA6A 37 0A             .BYTE   55,10
                ;
BA6C 5E 5F 82          .BYTE   94,95,130
                ;
BA6F 7E 23 A2          .BYTE   126,35,162,100
BA72 64
                ;
BA73 2E                .BYTE   46
                ;
BA74 95 EB 75          .BYTE   149,235,117
                ;
BA77 C3 ED             .BYTE   195,237
                ;
BA79 45 E7 B5          .BYTE   69,231,181
                ;
BA7C 33 E7             .BYTE   51,231
                ;
BA7E E6 E7 F6          .BYTE   230,231,246,200
BA81 C8
                ;
BA82 6E 70 82          .BYTE   110,112,130
                ;
```

```
BA85 33 E3              .BYTE   51,227
;
BA87 C5 E3 96           .BYTE   197,227,150
;
BA8A 48 E3 70           .BYTE   72,227,112,3,93
BA8D 03 5D
;
BA8F 6E 85 04           .BYTE   110,133,4
;
BA92 25 EF 80           .BYTE   37,239,128
;
BA95 53 E0              .BYTE   83,224
;
BA97 46 E0 4C           .BYTE   70,224,76,242
BA9A F2
;
BA9B 6E 04 73           .BYTE   110,4,115
;
BA9E C3 35              .BYTE   195,53
;
BAA0 C6 35 6C           .BYTE   198,53,108,125
BAA3 7D
;
BAA4 33 AB              .BYTE   51,171
;
BAA6 A3 2A 9E           .BYTE   163,42,158,55,165
BAA9 37 A5
;
BAAB BC 32 A4           .BYTE   188,50,164,86,0,251,83
BAAE 56 00 FB
BAB1 53
;
BAB2 53 28 24           .BYTE   83,40,36
;
BAB5 63 24 F3           .BYTE   99,36,243
;
BAB8 64 37 74           .BYTE   100,55,116
;
BABB 53 7F 6C           .BYTE   83,127,108
;
BABE 33 E6              .BYTE   51,230
;
BAC0 63 80 50           .BYTE   99,128,80
;
BAC3 13 3F              .BYTE   19,63
;
BAC5 63 89              .BYTE   99,137
;
BAC7 65 8B 7E           .BYTE   101,139,126
;
BACA 26 8D 96           .BYTE   38,141,150,248
BACD F8
;
BACE 58 EB F8           .BYTE   88,235,248
;
BAD1 58 09 34           .BYTE   88,9,52
;
BAD4 78 09 F5           .BYTE   120,9,245,50
BAD7 32
;
```

```
BAD8 58 84 2E           .BYTE   88,132,46
                        ;
BADB 38 47              .BYTE   56,71
                        ;
BADD 38 4E              .BYTE   56,78
                        ;
BADF 48 F0              .BYTE   72,240
                        ;
BAE1 63 8F              .BYTE   99,143
                        ;
BAE3 D3 45              .BYTE   211,69
                        ;
BAE5 65 43 5A           .BYTE   101,67,90
                        ;
BAE8 22 02              .BYTE   34,2
                        ;
BAEA 90 2A 05           .BYTE   144,42,5,0,3
BAED 00 03
                        ;
BAEF 60 2A 05           .BYTE   96,42,5
                        ;
BAF2 D5 02 20           .BYTE   213,2,32
                        ;
BAF5 95 02 00           .BYTE   149,2,0
                        ;
BAF8 24 02 B4           .BYTE   36,2,180
                        ;
BAFB 60 2D 25           .BYTE   96,45,37
                        ;
BAFE 64 02 62           .BYTE   100,2,98
                        ;
BB01 40 27              .BYTE   64,39
                        ;
BB03 39 02 84           .BYTE   57,2,132,204,32
BB06 CC 20
                        ;
BB08 4B 02 04           .BYTE   75,2,4,83,149,106
BB0B 53 95 6A
                        ;
BB0E 22 03              .BYTE   34,3
                        ;
BB10 60 35 3C           .BYTE   96,53,60
                        ;
BB13 8B 03 96           .BYTE   139,3,150,66,58,46
BB16 42 3A 2E
                        ;
BB19 46 03 D2           .BYTE   70,3,210,86
BB1C 56
                        ;
BB1D 90 38 4C           .BYTE   144,56,76,194,4
BB20 C2 04
                        ;
BB22 B0 30 45           .BYTE   176,48,69,57,86,162,0
BB25 39 56 A2
BB28 00
```

```
008E            STAR    =       142
005B            POWER   =       91
0028            FACTOR  =       40
0054            PERCEN  =       84
```

```
STO             119
STP             120
SUB             121
SUM             122
TAN             123
TRACE           124
X               125
XCHGY           126
XCHM            127
XEQ             128
XGE             129
XLT             130
XMEAN           131
XNE             132
XOR             133
XSTDDEV         134
XVARIANCE       135
Y               136
YARDS           137
YINTERCEPT      138
YMEAN           139
YSTDDEV         140
YVARIANCE       141
```

```
000C                    BSTEP   =       12
0000                    CLP     =       0
0024                    ZEND    =       36
005E                    PROGRAM =       94
0076                    SSTEP   =       118
0078                    STP     =       120
0039                    INSERT  =       57
0021                    DELETE  =       33

BB29                    PRIOTB

                        ;
BB29 DD EE E6                   .BYTE   221,238,230,237,222,221,238,221,221,238                 ACOS            0
BB2C ED DE DD
BB2F EE DD DD
BB32 EE
                        ;
BB33 ED DD ED                   .BYTE   237,221,237,237,221,221,238,221,238,221                 CLR             0
BB36 ED DD DD
BB39 EE DD EE
BB3C DD
                        ;
BB3D DD DE DD                   .BYTE   221,222,221,222,238,221,237,221,222,222                 FAHRENHEIT      0
BB40 DE EE DD
BB43 ED DD DE
BB46 DE
                        ;
BB47 DD DE DD                   .BYTE   221,222,221,218,221,234,238,222,238,222                 KM              0
BB4A DA DD EA
BB4D EE DE EE
BB50 DE
                        ;
BB51 E5 DE AD                   .BYTE   229,222,173,238,221,217,222,238,238,238                 OR              0
BB54 EE DD D9
BB57 DE EE EE
BB5A EE
                        ;
BB5B ED DD DE                   .BYTE   237,221,222,233,222,174,221,221,221,238                 RANDOM          0
BB5E E9 DE AE
BB61 DD DD DD
BB64 EE
                        ;
BB65 ED DD ED                   .BYTE   237,221,237,221,221,221,213,221,221,221                 SUB             0
BB68 DD DD DD
BB6B D5 DD DD
BB6E DD
                        ;
BB6F DD 88 77                   .BYTE   221,136,119,34,16                                       YVARIANCE       0
BB72 22 10
BB74                    JMPTBL

BB74 B9 A5 C0                   .WORD   SABSVA,SACOS,SADV,SALG,SALGN,SAND,SANNUI,SASIN,SATAN,SBAL
BB77 A5 F9 A5
BB7A 16 A6 1A
BB7D A6 1F A6
BB80 44 B0 24
BB83 A6 4C B4
BB86 6A B0
BB88 3D A6 5D                   .WORD   SBIN,SBITS,SBSTEP,SCALL,SCDEG,SCELSI,SCGRAD,SCHGSG,SCLINI,SCLMEM
BB8B A6 D2 AB
```

COLLEEN CALCULATOR, BY C SHAW

```
BB8E AA AD 82
BB91 AF EA A6
BB94 86 AF FE
BB97 A6 07 A7
BB9A 14 A7
BB9C 39 AC 32                .WORD   SCLPRO,SCLR,SCLX,SCM,SCMPND,SCOMPL,SCONTI,SCORRE,SCOS,SCRAD
BB9F A7 9E A2
BBA2 54 AF 48
BBA5 B0 38 A7
BBA8 15 AD 03
BBAB B3 9B B3
BBAE 8A AF
BBB0 5C A7 41                .WORD   SDCDEG,SDEC,SDEG,SDELET,SDIV,SDMS,SEND,SENTER,SEXPE,SEXPTE
BBB3 A7 46 A7
BBB6 FB AD 4B
BBB9 A7 54 A7
BBBC 24 AD 60
BBBF B0 BF A7
BBC2 C5 A7
BBC4 CB A7 FA                .WORD   SFACTO,SFAHRE,SFEET,SFIND,SFIX,SFLOZ,SFRACT,SFV,SFVDUE,SFVORD
BBC7 A7 48 AF
BBCA 64 B0 11
BBCD A8 74 AF
BBD0 44 A8 B0
BBD3 B0 4E B0
BBD6 52 B0
BBD8 78 AF 6E                .WORD   SGAL,SGM,SGOTO,SGRAMS,SHEX,SI,SINCHE,SINSER,SINTEG,SINTME
BBDB AF 48 AC
BBDE 6E AF 50
BBE1 A8 D8 B0
BBE4 44 AF 1C
BBE7 AE 55 A8
BBEA 75 A8
BBEC 62 AF 58                .WORD   SKG,SKM,SLB,SLIST,SLITER,SLN,SLOGTE,SLSHF,SMETER,SMILES
BBEF AF 6A AF
BBF2 8C AC 7C
BBF5 AF 7C A8
BBF8 82 A8 88
BBFB A8 40 AF
BBFE 50 AF
BC00 11 AF 95                .WORD   SMLOAD,SMOD,SMSAVE,SN,SNAUTI,SNOP,SNOTRA,SNWEIG,SOCT,SOFF
BC03 A8 1A AF
BC06 26 B1 5C
BC09 AF 39 AD
BC0C C9 AC B3
BC0F B3 BA A8
BC12 C3 A8
BC14 E8 A8 2A                .WORD   SON,SOR,SOZ,SPAUSE,SPERCE,SPI,SPLOAD,SPMT,SPOLAR,SPOP
BC17 A9 66 AF
BC1A DA AC 2F
BC1D A9 3C A9
BC20 E6 AE BF
BC23 B1 4A A9
BC26 7E A0
BC28 6A AF 76                .WORD   SPOUND,SPOWER,SPRD,SPRINT,SPROGR,SPSAVE,SPUSH,SPV,SPVDUE,SPVORD
BC2B A9 D3 A9
BC2E 66 AB F9
BC31 AC EF AE
BC34 9C A0 0F
BC37 B2 58 B0
```

```
BC3A 5C B0
BC3C DC A9 E5                   .WORD   SRAD,SRANDO,SRCL,SRECIP,SRECTA,SRESET,SRETUR,SROOT,SROUND,SRPN
BC3F A9 F8 A9
BC42 03 AA 08
BC45 AA 09 AD
BC48 E3 AD 37
BC4B AA 51 AA
BC4E 3D AA
BC50 5E AA 12                   .WORD   SRSHF,SRUN,SSIN,SSLOPE,SSMINU,SSPLUS,SSQRT,SSQUAR,SSSTEP,SSTO
BC53 AD AA B3
BC56 5A B3 88
BC59 B2 8C B2
BC5C BA B4 AB
BC5F AA 10 AC
BC62 B1 AA
BC64 24 AD D2                   .WORD   SSTP,SSUB,SSUM,STAN,STRACE,SX,SXCHGY,SXCHM,SXEQ,SXGE
BC67 AA DB AA
BC6A E7 AA CD
BC6D AC 2A B3
BC70 F9 AA 02
BC73 AB 4C AD
BC76 55 AD
BC78 60 AD B9                   .WORD   SXLT,SXMEAN,SXNE,SXOR,SXSTDD,SXVARI,SY,SYARDS,SYINTE,SYMEAN
BC7B B2 6B AD
BC7E 2D AB C1
BC81 B2 CD B2
BC84 18 B3 4C
BC87 AF 3C B3
BC8A BD B2
BC8C C7 B2 D1                   .WORD   SYSTDD,SYVARI
BC8F B2
BC90 89 AB 98                   .WORD   SPMUL,SPDIV,SPADD,SPSUB
BC93 AB A7 AB
BC96 B6 AB

                        *=$A000+$2000-6                 ;CARTRIDGE START INFO
BFFA 00 98                      .WORD   START           ;COLD/WARM START ADDRESS
BFFC 00 05                      .BYTE   0,4+1           ;BOOT DISK & RUN CARTRIDGE
BFFE E4 9B                      .WORD   INIT            ;POWER UP START VECTOR
                                .END
```

COLLEEN CALCULATOR, BY C SHAW

SYMBOL TABLE

AFP	D800	ALGNOP	0002	ALGP	0001	ANNFLG	0B68
ARCSUB	A49D	ASAVE	00AC	ASMBL	0000	ATAN1	B46F
ATAN2	B4A9	ATCOEF	DFAE	ATNOUT	B4B7	AUDC1	D201
AUDF1	D200	BACKSP	007E	BB05	A56D	BB10	A574
BB20	A580	BB30	A585	BBLP1	A563	BBLP2	A576
BBLP3	A587	BIMSG	B7A8	BIN10	9CAF	BIN100	A65C
BIN2	00C1	BIN20	9CE2	BINARY	00BD	BINC10	9C90
BINCHK	9C7A	BINER2	9C9E	BINERR	9CA0	BINFP	9CA3
BINFP2	9CA8	BINLP1	9C80	BINLP2	9C92	BINMIN	00B9
BINOK	9C8A	BITBIN	00B1	BITBN2	00B5	BITINT	00A7
BITMSG	B776	BLKBUF	0B00	BLNK16	A358	BLNKS	A35A
BOMSG	B736	BRKKEY	0011	BRKLST	ACB8	BSTEP	000C
BTSMSG	B839	C1PT8	B63C	C65536	B630	CALPTR	0B6E
CALSTK	0B80	CDGR	AF8C	CDTMF3	022A	CELMSG	B85A
CHRTAB	B6CC	CHS30	A326	CHSTAT	A30D	CHTAB2	B6D5
CIOCAL	AEAB	CIOV	E456	CIX	00F2	CLEMSG	B7D1
CLFMSG	B7DB	CLLP	A70B	CLNLP	9CE7	CLNUM	9CE3
CLOSE	000C	CLP	0000	CLRPTR	008C	CLRTAB	009E
CLS	007D	CN1MSG	B86F	CN2MSG	B87B	CN3MSG	B886
COLCMD	0014	COLCRS	0055	CONERR	AFC3	CONFLG	00D2
CONV10	AFAF	CONVRN	AFC2	CONVRT	AF8E	CR	009B
CRSINH	02F0	CRYCHK	ABBC	CRYCLR	ABC7	CRYMSG	B79B
CRYSND	ABBE	CTLR	A352	CTLR17	A350	CTLRS	0B26
CURPRI	00A3	DALG	0006	DBITS	0018	DDEC	0010
DDEG	000B	DEGREE	B690	DEGSUB	A762	DELCHR	00FE
DELETE	0021	DELLIN	009C	DFIX	001E	DG10	9DA2
DG20	9DBF	DG30	9DD1	DG60	9DEF	DG70	9E07
DG80	9E09	DGLP1	9DB2	DGLP2	9DBD	DGLP3	9DCB
DGRTN	9E14	DHO10	A17C	DHO20	A186	DHOCHK	A16C
DHOERR	A19B	DHOFLG	0087	DHOOK	A196	DIGMSG	B782
DIGRT	00F1	DISP	00A6	DIVI	B018	DM10	9D50
DMCLP	A728	DMCLR	A71B	DMELP	9D1B	DMEMAL	9D14
DMFLG	0B6B	DNAROW	001D	DOFF	0020	DOLOP	AB2F
DSPCOM	9CEE	DSPFLG	00CA	DSPM2	9D30	DSPM3	9D47
DSPM4	9D46	DSPMEM	9D2E	DSPRG	9D97	DSPSTK	9D51
DST10	A6D8	DSTAT	A6B9	DSTLP	A6CD	DUEFLG	0B67
EDSKP2	9A5F	EEXP	00ED	ENDL10	9B1E	ENDL20	9B27
ENDLP	9A48	ENDLP3	9A4C	ENDLP4	9A56	ENDLST	9B13
ENDSKP	9B1C	ENDWLP	9A41	ENDWRD	9B3A	ENTFLG	0B69
EPERR	9D9D	EPMSG	B7BC	EQUAL	0094	ERRFLG	0B6C
ERRMSG	B6C1	ERRSB2	9C13	ERRSUB	9BE5	ERRTBL	B700
ESC	001B	ESIGN	00EF	EXEC	0002	EXEC10	9A8E
EXEC20	9A94	EXP	DDC0	EXP10	DDCC	FABCD	A01C
FACTOR	0028	FADD	DA66	FAHMSG	B866	FASC	D8E6
FCHRFL	00F0	FCLOSE	AE48	FDIV	DB28	FDLP2	9EC4
FDLP3	9EAF	FDLP4	A00A	FDS05	9EE7	FDS1	9ED7
FDSCOM	9E5D	FDSP0	9E73	FDSP1	9E92	FDSP2	9E97
FHALF	DF6C	FIXFLG	009B	FIXMSG	B81D	FIXNUM	00CE
FLD0M	A034	FLD0P	DD8D	FLD0R	DD89	FLD0S	A03B
FLD0T	A047	FLD1P	DD9C	FLD1R	DD98	FLD1S	A041
FLD1T	A04E	FLPTR	00FC	FMOVE	DDB6	FMOVE2	A06F
FMUL	DADB	FMVPOP	A07B	FOP10	AE61	FOP20	AE8B
FOP30	AE8F	FOPEN	AE57	FOPLP1	AE68	FOPLP2	AE74
FOPLP3	AE6B	FP10	9E1F	FP15	9E25	FP20	9E2C
FP9S	DFEA	FPBIN	9E15	FPBNCK	9E2F	FPC10	A0C2
FPERR	9E58	FPI	D9D2	FPOP	A084	FPOP0	A07E
FPOP1	A075	FPOP10	A098	FPOPLD	A08A	FPPTR	00A4
FPREC	0006	FPSCR	05E6	FPSCR1	05EC	FPSH05	A0A2
FPSHLD	A0B3	FPSLEN	002A	FPSTK	0600	FPTR2	00FE

COLLEEN CALCULATOR, BY C SHAW

FPUSH0	A09C	FPUSH1	A0AA	FPX	0B5E	FR0	00D4
FR1	00E0	FR2	00E6	FRE	00DA	FRX	00EC
FSCR	05E6	FSCR1	05EC	FSLOP	A066	FSPMSG	B951
FSQR	B4E5	FSTOP	DDAB	FSTOR	DDA7	FSTOT	A055
FST1R	A060	FST1T	A05C	FSUB	DA60	FTEMP	0B52
FTEMP2	0B58	GETC05	A0F3	GETC06	A12D	GETC07	A139
GETC10	A143	GETC12	A14B	GETC15	A14F	GETC20	A156
GETC30	A15B	GETCHR	0007	GETDH0	A160	GETINT	A19E
GETMN	A4B1	GETPRI	A1D2	GI05	A1B4	GI10	A1C6
GI20	A1C8	GINT2	A1BD	GMERR	A4C4	GNOCR	A11F
GPR10	A1DD	GPR20	A1EF	GTCHR	A0E8	ICAX1	034A
ICAX2	034B	ICBAH	0345	ICBAL	0344	ICBLH	0349
ICBLL	0348	ICCOM	0342	ICDNO	0341	ICHID	0340
ICPTH	0347	ICPTL	0346	ICSPR	034C	ICSTA	0343
IFP	D9AA	INBUFF	00F3	IND10	A1F7	INIT	9BE4
INIT2	9803	INIT3	A3D1	INIT4	9845	INPUT	0004
INS10	AE35	INS30	AE41	INSCHR	00FF	INSERT	0039
INSLIN	009D	INTADD	ABA1	INTC2	A295	INTCMP	A28D
INTDIV	AB92	INTDSP	A1F0	INTERM	00A0	INTFLG	00AA
INTLBF	DA51	INTMOD	A88D	INTMUL	AB83	INTSUB	ABB0
IOCB	0340	IOCBSZ	0010	IOELP	AECE	IOERR	AEB4
ITER	0B6A	JMPTBL	BB74	JMPTR1	0095	JMPTR2	0097
KBUFF	B6BE	KEY06	9AD9	KEY07	9AF0	KEY10	9B02
KEY20	9B07	KEY25	A3F4	KEY30	A3FB	KEY40	A42C
KEY50	A42E	KEY60	A438	KEY70	9B10	KEYCHR	0088
KEYCNT	008F	KEYERR	9B17	KEYLEN	0089	KEYLN2	008A
KEYLP1	9AC6	KEYLP2	9ACB	KEYMSG	B70D	KEYWRD	B964
KIOCB	0010	KLRSV	0092	KMATCH	0094	KYCNSV	0093
KYLFRT	008E	KYPTSV	0090	LBPR1	057E	LBPR2	057F
LBUFF	0580	LDBAL	AFC8	LDCH05	A21F	LDCH10	A222
LDCHR	A1FC	LDCSAV	0B64	LDFV	AFCC	LDI	AFD0
LDINT	A2A2	LDN	AFD4	LDN20	A237	LDNBSV	008B
LDNIB	A227	LDPMT	AFD8	LDPV	AFDC	LENERR	9BC6
LENG	AF5E	LENGTH	B642	LEX	9A79	LEX02	9ABC
LEX05	9ABF	LEX30	9AB5	LEXERR	9BC8	LEXRTN	9BD8
LFAROW	001E	LFRT	0080	LINLEN	0026	LMARG	0001
LMARGN	0052	LOG	DECD	LOG10	DED1	LOOP	98BC
LOOP3	98EA	LOOP4	9909	LOP10	AB52	LOP20	AB5B
LOP30	AB5D	LOPLP1	AB36	LOPLP2	AB47	LPAD	0095
LPAR	0092	LX50	9B94	LX60	9B9F	LXERR2	9B59
LXGT2	9BA7	LXHVDT	9B72	LXINIT	9A62	LXLP20	9AA5
LXLP40	9B5D	LXN2	9BB4	LXND2	9B7D	LXNDOT	9B54
LXNMCK	9B41	LXNUM	9BB1	LXRTN2	9BDE	LXSAV0	9BD6
LXSAV1	9BCE	MAIN02	98D4	MAIN03	98DD	MAIN04	9978
MAIN05	9981	MAIN10	A0DC	MAIN15	9CF9	MAIN20	9CFF
MAIN21	9D13	MAIN35	99A1	MAIN40	99A4	MAIN50	99B4
MAIN60	99BF	MAIN65	99CB	MAIN70	99DA	MAIN76	99DC
MAIN80	99E8	MANTLN	00CF	MAS	AF70	MASS	B672
MATCH	AD80	MEMA10	B253	MEMA20	B259	MEMADD	B245
MEMDIV	B27D	MEMFLG	0B6D	MEMLD0	A4C9	MEMLD1	A4CF
MEMLD2	A4D5	MEMLDR	A4D7	MEMLEN	0064	MEMLOD	AFDE
MEMMSG	B845	MEMMUL	A4FE	MEMNUM	00AF	MEMORY	0800
MEMST0	B071	MEMSUB	A4E7	MINUS	0091	MLD10	A4E1
MLD20	A4E5	MODFAC	0B4C	MS10	A4F0	MSAV10	AF21
MSAV15	AF3A	MSAV20	AEE5	NATCF	000B	NCHKLD	A23C
NCK10	A253	NCK30	A260	NCK40	A261	NEGFLG	00A9
NERR	AC06	NODMSG	B72A	NOEXEC	9A95	NOMAT	AD72
NOMMSG	B717	NOPFLG	009F	NOPMSG	B78E	NOSNER	B3C7
NOST02	995B	NOSTOR	994C	NSCF	0006	NSEFLG	009A
NSEMSG	B747	NSFMSG	B753	NSIGN	00EE	NUMBER	0096

COLLEEN CALCULATOR, BY C SHAW

NUMFLG	00AB	NUMLEN	0010	OFFERR	A909	ONESUB	ABC8
OPEN	0003	OPFLG	009E	OPNIN	AE51	OPNOUT	AE55
OPPTR	00A5	OPSLEN	0100	OPSTK	0700	OSEMSG	B75E
OSFMSG	B76A	OUTPUT	0008	PAND	0006	PBUFF	B6BB
PC	00C6	PC1MAX	000F	PCADD	A27C	PCADD1	A28A
PCADDN	A27A	PCINC	A276	PCLR0	A29E	PCN05	A26E
PCN10	A271	PCN20	A275	PCNCHK	A262	PEQUAL	0001
PERCEN	0054	PHIGH	000D	PICONS	B600	PIOCB	0020
PIOV18	B636	PIOV4	DFF0	PIOVL	A509	PKPTR	008C
PLPAD	0000	PLRPAR	0002	PLUS	0090	PLYARG	05E0
PLYEVL	DD40	POP10	A2BA	POPC10	AD8F	POPCAL	AD84
POPN10	A911	POPN20	A928	POPOP	A2B0	POR	0005
POWER	005B	PPLUS	0007	PPOWER	0009	PREVOP	00A1
PRGLEN	0400	PRGMEM	0C00	PRIOTB	BB29	PRMFLG	009C
PRNCHK	A3AE	PRNFLG	009D	PROG	00C8	PROGRA	005E
PROMSG	B800	PRVPRI	00A2	PRVSTK	00B0	PSAV10	AEF6
PSET0	A2A7	PSH10	A2CF	PSHC10	ADA2	PSHCAL	AD97
PSPEC	000E	PSPEC2	000A	PTABC	A2FC	PTABD	A370
PTCHR	A2F2	PTCHR2	A2F4	PTCHS2	A366	PTCHSP	A393
PTCRPN	A381	PTDEL2	A2D6	PTIMES	0008	PTLIN1	A331
PTLP	987A	PTMSG2	9C6C	PTTXTP	A3BA	PUSHOP	A2C1
PUTBLK	A023	PUTBRT	A033	PUTCHR	000B	PUTCHS	A35E
PUTCMD	9DF6	PUTCR	A2DE	PUTCRP	A385	PUTDEL	A2D2
PUTMSG	9C6A	QUADFL	00C5	RADFLG	00FB	RADPI2	B624
RAMCLR	A3C4	RAMSET	A3C6	RANDOM	D20A	RETN1	A2F1
RETN2	A3B9	RETURN	A5BF	RF10	9F17	RF100	9F62
RF105	9F7B	RF110	9F84	RF120	9FA0	RF130	9FB1
RF140	9FB7	RF142	9FD6	RF148	9FDF	RF150	9FEB
RF160	9FFA	RF170	A002	RF20	9F1F	RF30	9F21
RF40	9F2B	RF50	9F30	RF70	9F12	RF80	9F3B
RF90	9F5F	RFERR	9F6D	RFLP1	9F43	RFLP2	9F64
RFLP3	9F92	RFLP4	9FA2	RFLP5	9FF3	RMARG	0026
RMARGN	0053	ROWCMD	0016	ROWCRS	0054	ROWREG	0005
ROWSCR	0010	ROWSTT	0001	RPAR	0093	RPNALG	0099
RTAROW	001F	S180PI	A5D8	S2CMP	A51E	SABSVA	A5B9
SAC10	A5C4	SAC30	A5CA	SAC31	A5DF	SAC34	A5E4
SACOS	A5C0	SADV	A5F9	SADV10	A608	SALG	A616
SALG10	A61C	SALGN	A61A	SAND	A61F	SANNUI	B044
SAS10	A637	SASIN	A624	SATAN	B44C	SAVCHR	A3DF
SAVMAT	A3EA	SBAL	B06A	SBAL05	B06F	SBAL10	B07B
SBAL15	B080	SBAL20	B085	SBCL5	A0CE	SBERR	A678
SBIN	A63D	SBITO	A674	SBITS	A65D	SBITS2	A67D
SBLP1	A692	SBLP2	A69E	SBLP3	A6B0	SBSTO5	ABE3
SBST10	ABE5	SBST30	ABFD	SBST40	AC0B	SBST50	AC0F
SBSTEP	ABD2	SCAL10	ADC2	SCAL20	ADCC	SCAL30	ADE2
SCALL	ADAA	SCDEG	AF82	SCELSI	A6EA	SCGRAD	AF86
SCH10	A706	SCHGSG	A6FE	SCLINI	A707	SCLMEM	A714
SCLP2	A514	SCLPRO	AC39	SCLR	A732	SCLSTK	AA46
SCLX	A29E	SCM	AF54	SCMP10	B04A	SCMP2	A512
SCMPND	B048	SCOEF	B606	SCOMPL	A738	SCONFG	00D3
SCONTI	AD15	SCORRE	B303	SCOS	B39B	SCRAD	AF8A
SDCDEG	A75C	SDEC	A741	SDEG	A746	SDELET	ADFB
SDELP1	AE03	SDIV	A74B	SDMS	A754	SEFORM	00CD
SEND	AD24	SEND10	AD35	SENTER	B060	SETMSG	9C4F
SETTAB	009F	SETVBV	E45C	SEXPE	A7BF	SEXPTE	A7C5
SF10	A7DC	SFACTO	A7CB	SFADD	ABAA	SFAHRE	A7FA
SFDIV	AB9B	SFDON	A7F9	SFEET	AF48	SFERR	A7F4
SFIND	B064	SFIX	A811	SFIX2	A823	SFLOZ	AF74
SFLP	A7E0	SFMUL	AB8C	SFND10	B066	SFRACT	AB44
SFSUB	ABB9	SFV	B0B0	SFV05	B0B5	SFV10	B0BA

COLLEEN CALCULATOR, BY C SHAW

SFV20	B0CA	SFVDUE	B04E	SFVORD	B052	SFXERR	A83F
SGAL	AF78	SGM	AF6E	SGOERR	AC81	SGOTO	AC48
SGRAMS	AF6E	SHEX	A850	SHF05	AA70	SHF10	AABC
SHF15	AA8F	SHF20	AA9C	SHF30	AAAA	SHFSUB	AA60
SI	B0D8	SI05	B0E8	SI10	B0FE	SI20	B121
SIGSUB	B28E	SINCHE	AF44	SINDON	B43E	SINERR	B3C4
SINF1	B3F8	SINF3	B40B	SINF4	B436	SINFIN	B44B
SINMOD	B38A	SINSER	AE1C	SINSLP	AE24	SINT10	A864
SINT30	A868	SINT40	A86D	SINTEG	A855	SINTME	A875
SINTRT	A874	SIOCB	0000	SKG	AF62	SKM	AF58
SLASH	008F	SLB	AF6A	SLIST	AC8C	SLITER	AF7C
SLN	A87C	SLOGTE	A882	SLS05	A534	SLSHF	A888
SLSHF2	A530	SLST05	ACB1	SLST07	ACB4	SLST10	ACBF
SLSTLP	AC96	SMETER	AF40	SMILES	AF50	SMLOAD	AF11
SMOD	A895	SMSAVE	AF1A	SMSD	00D1	SN	B126
SN05	B12B	SN10	B130	SN20	B150	SN50	B17F
SNAUTI	AF5C	SNDLP1	9C31	SNOP	AD39	SNOTRA	ACC9
SNUM20	A54E	SNUM25	A593	SNUM30	A5A3	SNUM40	A5A9
SNUM50	A5B8	SNUMB	A543	SNWEIG	B383	SOCT	A8BA
SOCT10	A8BC	SOFF	A8C3	SON	A8E8	SOR	A92A
SORD10	B054	SOZ	AF66	SPADD	ABA7	SPAULP	ACF3
SPAUSE	ACDA	SPCEND	B6BB	SPCLEN	0006	SPCTBL	B6B4
SPDIV	AB98	SPE10	A93B	SPERCE	A92F	SPI	A93C
SPI10	A943	SPLOAD	AEE6	SPMT	B1BF	SPMT05	B1C4
SPMT10	B1C9	SPMT20	B1D1	SPMT30	B1EF	SPMUL	AB89
SPOLAR	A94A	SPOP	A07E	SPOUND	AF6A	SPOW10	A986
SPOW20	A98F	SPOW30	A99C	SPOW40	A99F	SPOW50	A9BC
SPOW60	A9D2	SPOWER	A976	SPRD	A9D3	SPRINT	AB66
SPROGR	ACF9	SPRRTN	AD08	SPSAVE	AEEF	SPSUB	ABB6
SPUSH	A09C	SPV	B20F	SPV05	B214	SPV10	B219
SPV20	B229	SPVDUE	B058	SPVORD	B05C	SQR0	B4CA
SQR1	B4DD	SQR2	B556	SQR3	B55C	SQRDON	B539
SQRLP	B4FF	SQROUT	B564	SRAD	A9DC	SRAD10	A9DE
SRAN10	A9EC	SRANDO	A9E5	SRCL	A9F8	SRCL10	A9FB
SRCL20	AA02	SRECIP	AA03	SRECTA	AA08	SRESET	AD09
SRET10	ADF8	SRET20	ADFA	SRETUR	ADE3	SROOT	AA37
SROUND	AA51	SRPN	AA3D	SRPN10	AA3F	SRSHF	AA5E
SRTN	A541	SRUN	AD12	SRUN10	AD1D	SSIGN	00D0
SSIN	B3AA	SSIN2	B3AD	SSLOPE	B35A	SSMINU	B288
SSPLUS	B28C	SSQRT	B4BA	SSQUAR	AAAB	SSSTEP	AC10
SSTEP	0076	SSTFLG	00CB	SSTO	AAB1	SSTO10	AAB6
SSTO12	AACB	SSTO14	AACE	SSTO20	AAD1	SSTOLD	00CC
SSTP	AD24	SSTP10	AC1D	SSTP15	AC33	SSTP20	AC36
SSUB	AAD2	SSUM	AADB	SSUM10	AAE1	STAN	AAE7
STAR	008E	STARMS	B6C8	START	9800	STATLN	B8FD
STCRTN	A52F	STKD10	9D61	STKD30	9D80	STKD40	9D91
STKD45	9D92	STKLIN	B937	STOPRG	0001	STP	0078
STPR40	9934	STPR50	993B	STPR51	9946	STR10	ACCF
STR20	ACD9	STRACE	ACCD	SUBCAL	A0C5	SUBONE	ABCC
SUCCES	0001	SX	B32A	SX10	AB2A	SXCHGY	AAF9
SXCHM	AB02	SXEQ	AD4C	SXGE	AD55	SXLT	AD60
SXMEAN	B2B9	SXNE	AD6B	SXOR	AB2D	SXPRIM	AB65
SXR10	AB75	SXRTN	AB82	SXSTDD	B2C1	SXVARI	B2CD
SY	B318	SYARDS	AF4C	SYINTE	B33C	SYMEAN	B2BD
SYSTDD	B2C7	SYVARI	B2D1	TO	00A8	TAB	007F
TABLE	B6DF	TIOCB	0030	TOKBUF	0500	TOKCHR	B69C
TOKCLN	000D	TOKCOD	0081	TOKEND	B6AA	TOKLEN	0026
TOKNUM	9EA0	TOKPTR	0082	TOKTBL	B6AA	TOKTIN	0086
TOKTMP	0084	TOPMSG	B700	TRACE	00C9	UKERR	A498
UNIMSG	B7C9	UNKRTN	A484	UNKY10	A476	UNKY20	A48D

COLLEEN CALCULATOR, BY C SHAW

UNP10	A44A	UNPACK	A43F	UNPCK2	A43B	UNPINT	A3FC
UNPKEY	A461	UNPLP	A479	UNPNUM	A40F	UNPNXT	A41B
UPAROW	001C	VOL	AF7E	VOLUME	B684	WLOOP	99EC
WLP05	9A0C	WLP10	9A15	WLP20	9A1F	WLP30	9A30
XEFORM	D920	XFORM	DE95	XLTERR	AD83	XLTSUB	AD3A
XSAVE	00AD	XSAVE2	0B65	YSAVE	00AE	YSAVE2	0B66
Z11IMN	B027	Z11INI	B02D	Z1IMN	B009	Z1IN	AFFD
Z1IN1	AFEE	Z1IN1I	B015	Z1INM1	B00F	Z1PLI	AFE6
ZDCDEG	B89E	ZDEG	B8F8	ZDMS	B891	ZEND	0024
ZFL	B8F0	ZKG	B8E9	ZLN1I	B021	ZM	B8E5
ZMRTN	B043	ZMUL1I	B033	ZPOLAR	B8C3	ZRECT	B8A5
ZROPG	0080	ZSIGMA	B267	ZTEMP1	00F5	ZTEMP3	00F9
ZTEMP4	00F7	ZVAR	B2D3	ZVAR2	B2E1		